Mountain Plains States Consortium
WIC System Project

DETAILED TECHNICAL SPECIFICATIONS DOCUMENT

Presented to:

MPSC WIC

WY

UT CO

Revision Date: Final – March 24, 2008

Prepared by

ciber

650 Wilson Lane, Suite 200
Mechanicsburg, PA 17055
717.691.5500

CO Contract #WIC0601052
CIBER Project #CODPH00201

Document Revisions

Revision Date	Updated By	Requested By	Description of Revision
1/8/07	CIBER	MPSC	Revision Version
1/15/08	CIBER	MPSC	Final Version
3/24/08	CIBER	MPSC	Updated Final Version

Table of Contents

1 Introduction

This document describes the architecture and technical design for the Mountain Plains States Consortium (MPSC) system, including a description of the overall architectural approach and the reasoning and motivation behind the decisions leading to the selected architecture. Where applicable, this document also describes how the system has been designed to evolve over time.

The architectural approach for the MPSC system is based on the Smart Client architecture published and supported by Microsoft. The specification from Microsoft is based on a Service-Oriented Architecture, in which individual systems are implemented as interoperating sets of services, using common frameworks, standards, and practices.

The architectural solution will enable the MPSC system to become a participant system in the overall enterprise application space across the states and will be classified as a “Tier II” system. A Tier II system is defined as a software system targeted at a specific set of business users, as opposed to being a Tier I service available to everyone, such as a Shared Security Service. The design methodology sets out a software development process focusing on the systemic qualities and early identification and mitigation of risk.

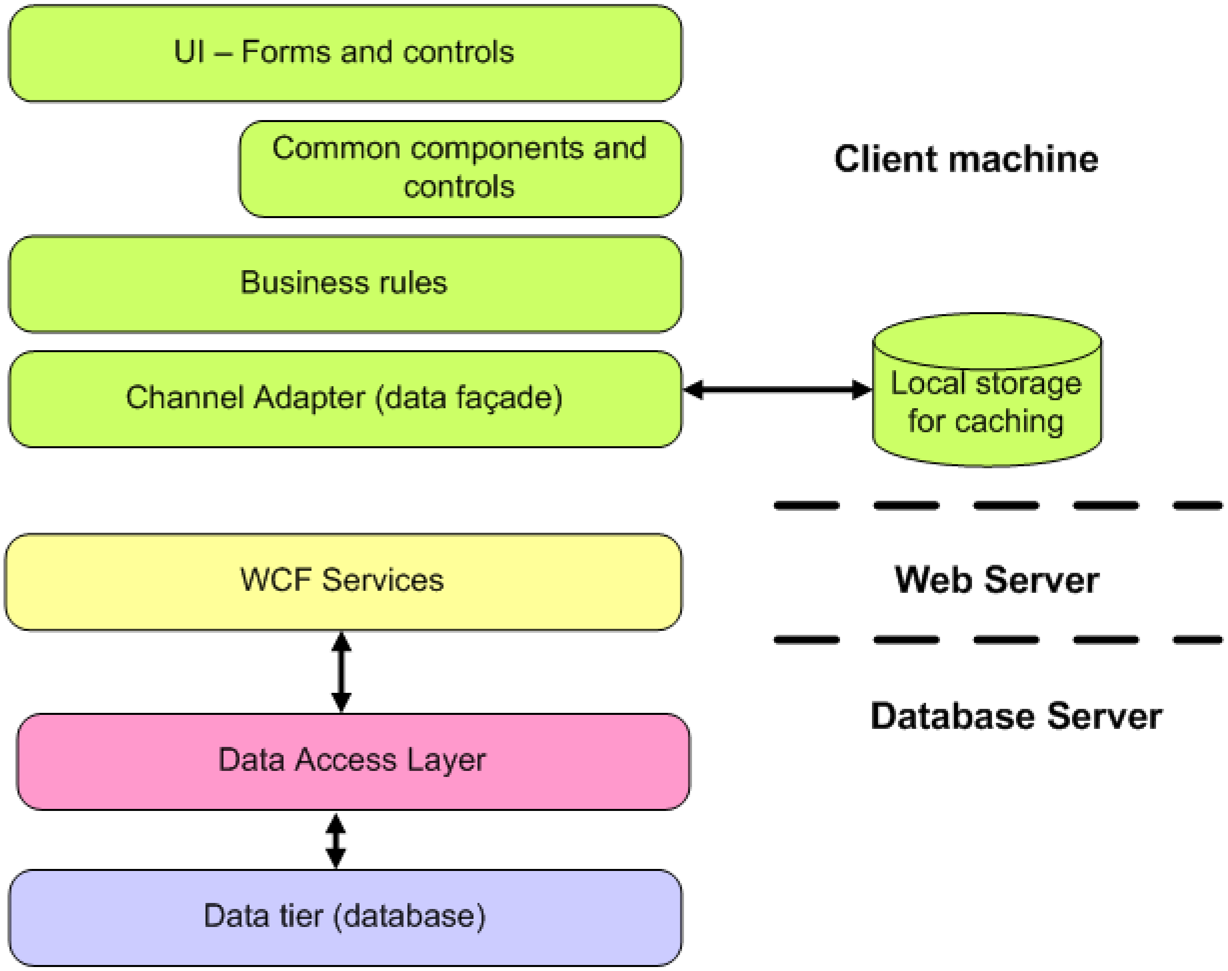
This document outlines the various parts of the architecture and then drills down into the details on the design of those areas. Wherever applicable, there are references to APIs and frameworks that must be considered transient at this point since this document will be updated before the final delivery of the system to MPSC.

2 Architectural Overview

The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them. In addition, the software architecture work products define interactions which take place between these components to realize a business process.

The software architecture model below describes the overall design of the MPSC system in terms of the software system, detailing the various application layers that make up the design. The solution uses an n-tier approach for building the MPSC system, utilizing best practices of software application partitioning in order to achieve the design and performance goals of the system. This n-tier approach was specifically designed and tailored for the MPSC project.

The n-tier architecture for the MPSC is further specialized using the framework provided by the "Smart Client Architecture". This is based on the premise of handling all major processing and business steps on or as close to the client as possible. The diagram below illustrates the logical architecture of a distributed Smart Client system.



2.1 Client Machine

The client machine contains the major part of the working application in compiled executables, DLLs (Dynamic Link Libraries), and supporting files. In addition to the user interface, the client machine also provides the following functions:

- Performs basic validation of data captured through user interface interaction.
- Applies business rules to the data captured through the user interfaces, which may span data collected across multiple screens.
- Caches information to ensure minimal communication with the server, and hence, optimal use of the network.
- Applies security to the various parts of the user interface.

The client for the system uses the Windows Forms interface, and hence, all controls used during the development are native to the .NET framework. The guiding principles for the development of the application for the client machine are:

- Develop controls based on well-defined component architecture.
- Minimize repeated code patterns in routine forms with frameworks such as:
  - Data validation
  - Data management
  - Security
- Retrieve information from the server when required. In context of the user interface for a connected system, this translates to fetching information for a specific "unit of work" when requested.
- Compress the communication between the client and Web server to ensure optimal use of network bandwidth.

The above discussion also applies to a disconnected client system where the communication with the Web server happens in batch for synchronization.

2.2 Web Server

The Web server provides all communication interfaces back to the central database. The Web server provides the following high level functions:

- Communication end-point for all interaction between client and server.
- Compression for all packets to and from the client.

The guiding principles for the design and development of the Web server are:

- Build interfaces based on specialization to ensure scalability.
- Ensure secure communications between client and server.
- Use abstraction to ensure that the contract between client and server can adapt to changes in the actual implementation of the contract on the server.

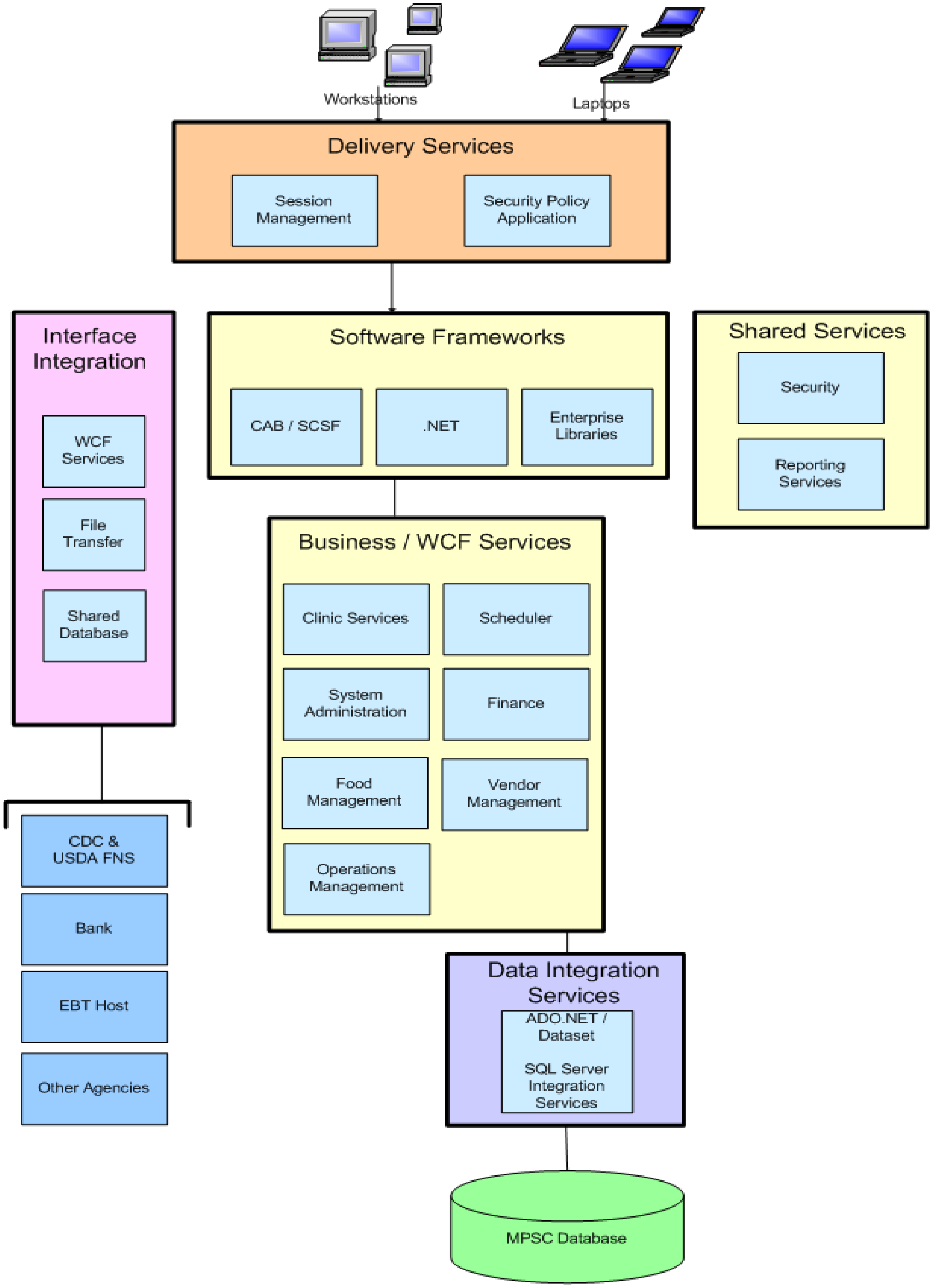
The Web server provides a very specialized service within this architecture, specifically facilitating communication between client and server. Using the right deployment and configuration, this is also one of the key parts of the architecture that provides scalability to the solution. This is detailed in the System Deployment section.

2.3 Database Server

The database server provides the persistent data store for all data within the system. There are multiple physical data stores (databases) within the MPSC system that are detailed later in the document. The basic guiding principle for the On-Line Transaction Processing (OLTP) database is to ensure that data is persisted and retrieved in the most expedient fashion while maintaining referential integrity throughout a properly normalized logical data model. Also discussed later is high-level database infrastructure related to availability, operational continuity, scalability, and performance.

3 System Architecture

The following diagram shows the overall architecture. Sections are detailed below.



3.1 Delivery Services

Delivery services provide a consistent interface to the system for the users, regardless of device type or user roles. Their general role is to coordinate users' requests through accessing other services. Delivery services are accessed via the Internet or the intranet using HTTP(S) (Hyper-Text Transfer Protocol, Secured) and XML (eXtensible Markup Language).

Security Policy Application

These services access security services for both authentication and authorization of user requests, primarily involving the security shared service.

Session Management

Session management services maintain user session information, such as security roles, single sign on data, and other data that must be retained for a user session between requests.

3.2 Business Services

MPSC business services contain the rules used to accomplish WIC business-related tasks. These services are orchestrations of lower-level services, wherever possible, to maintain the architectural flexibility. The services are deployed for internal and external access so that both disconnected and connected clinic operations function using the same code base. Business services provide basic functionality needed by almost all components across the application. .NET Remoting provides efficient access to these internal services; however, for ease of maintenance and consistency, these internal services use Windows Communication Foundation (WCF) Services. The WCF services in connected mode are available with wsHTTPBinding, whereas the data is transported over HTTPS and encoded in either text or Message Transmission Optimization Mechanism (MTOM).

Clinic Services

Clinic services encapsulate all functionality necessary to bring new WIC families into the program, as well as to re-certify existing participants. The service also includes functionality for providing WIC benefits either through paper FIs or EBT cards depending on the state's preference.

Scheduler

The scheduler is a set of services that allows users to manage appointments for WIC families.

System Administration

These services enable the authorized user to modify data that is required for the MPSC system to work properly. This service is split into 5 services – Clinic Services, Scheduler, Vendor Management, Operations, and System-Wide services.

Finance

These services are used to provide support in the financial area for such things as grant management and formula rebates.

Food Management

These services are used to define WIC approved foods, participant profiles, food rules, and model food packages.

Vendor Management

These services are used to provide support in the authorization and management of businesses that apply for and are authorized to participate in the state WIC program.

Operations Management

These services are used for the purpose of providing support in the operations area for such things as inventory and staff support.

3.3 Shared Services

Shared services span functional areas (business services), providing standard implementation of these aspects system-wide.

Security

The security services provide authentication and authorization capabilities to the MPSC system. Security is role-based and roles are defined based on users' accessibility to business objects and/or tasks with respect to state, local agencies, and clinics.

Reporting Services

Reporting services are used by the MPSC system architecture. They provide the capability to access data for reporting. Reporting functionality uses the Microsoft SQL Server Reporting Services (SSRS) toolset. Ad hoc reporting is available using SQL Server Report Modeler, whereas the managed reports are designed using Report Designer.

3.4 Software Frameworks

Frameworks are used throughout the MPSC system to provide a variety of software functionality. The use of hardened, commercially-available frameworks enable developers to focus on solving the business problems by implementing with existing, proven frameworks and to avoid re-invention of infrastructure upon which application components are built.

Enterprise Library

The Enterprise Library is a collection of application blocks that the MPSC development team adopts to expedite development. The MPSC development team uses the Caching Application Block, Exception Handling Application Block, Logging Application Block, and Validation Application Block.

Smart Client Software Factory (SCSF)

MPSC system is designed using the SCSF framework where the components are designed independently and loaded dynamically and integrated together at runtime within a shell. MPSC system is designed and developed by applying use-case driven strategy recommended by the SCSF.

3.5 Interface Integration Services

These services form a two-way communications path between the WIC environment and external entities. Based on the requirements, either WCF services or XML Web services are exposed to the external entities as the standard means of access. Note that some file transfer interfaces are also provided as needed by this service grouping, though these are minimized as much as possible.

WCF Services

This service is used wherever the data is transported between the MPSC system and any .Net-based external component (e.g., Card Reader Interface System - CRIS).

File Transfer

Some external systems may require the transfer of files for the transmission of data. This is most commonly done via File Transfer Protocol (FTP) but may also be done through the use of common file shares if the external system has access to the same domain as the MPSC system. While the file formats have been determined during design, some of the implementation decisions for certain interfaces will be made later in the project.

3.6 Data Integration Services

Persistent storage must be accessed and updated, whether these stores are relational databases or legacy systems. The data integration services enable consistent access to these data stores.

ADO.NET (ActiveX Data Objects)

This is the standard access method for .NET-based systems to interface with relational databases. These services abstract the .NET calls, providing a consistent code-level interface.

SQL Server Integration Services (SSIS)

These services are provided by Microsoft SQL Server 2005 for developing interfaces to and from the database. The services are exposed as a set of native scripts and plug-ins available with the product. The solution leverages these services for creating extracts and database loads.

Logically, the integral parts of this system are described in the diagram in Section 3; however, the overall application environment for the MPSC system is Microsoft.NET. The framework provides these low-level services:

- Transaction Management
- Multi-threading
- Resource Pooling

3.7 External Interfaces

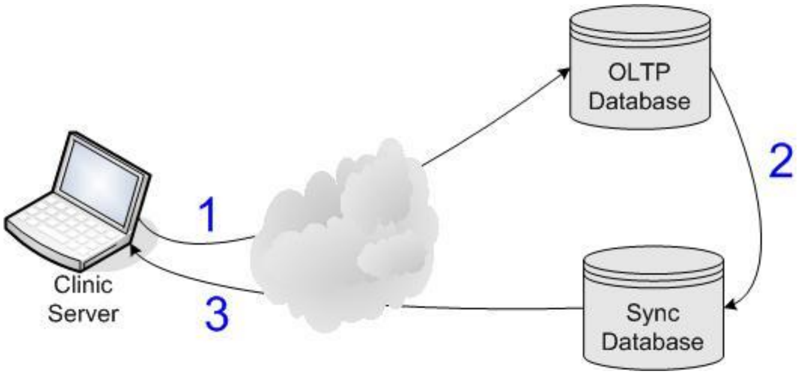
The table below lists examples of the interfaces external to the MPSC application with which the application exchanges data. All interfaces are detailed in the DFDD.

External Interface	Data Description	Format	In/Out	Frequency	Method / How
Auditor	Auditor's File	Fixed length (ASCII)	Out	As needed	TBD
Auto-Dialer (Utah Only)	Schedule data	Fixed length (ASCII)	In/Out	Daily	File Share
CDC	Pediatric Nutrition Surveillance (PedNSS) and Pregnancy Nutrition Surveillance (PNSS) data	Fixed length (ASCII)	Out	PNSS = Quarterly PedNSS = Monthly	Internet via CDC's Secure Data Network
EBT Host (WESS)	Approved Foods, Hot Cards, Pricing, and Redemption data	Fixed length (ASCII)	In/Out	Daily	WCF – encoded text
Formula Company	Formula Rebate File	Fixed length (ASCII)	Out	As needed	TBD
FSMC	Issuance, Vendor, and Redemption data	Fixed length (ASCII)	In/Out	Daily	FTP
Other States	Dual Participation data	Fixed length (ASCII)	In/Out	As needed	TBD
PRAMS	Pregnancy Risk Assessment Monitoring System (PRAMS) data	Fixed length (ASCII)	In/Out	As needed	FTP
USDA UPC National Database	UPC Product Information	Fixed length (ASCII)	In/Out	As needed	Secure connection
USDA-FNS	PC2008 and TIP Report	Fixed length (ASCII)	Out	PC2008 = Biennially TIP = Annually	FTP
USIIS (Utah Only)	Immunization criteria data	Fixed length (ASCII)	Out	Monthly	FTP
Vendor Portal (Utah Only)	Vendor Price Survey Data	Fixed length (ASCII)	In/Out	As needed	TBD

The external interfaces are handled using SQL Server Integration services. The outputs for the extracts are propagated to separate tables, which are then used to transmit information to the various agencies.

3.8 Disconnected Operations

The WIC application supports both connected and disconnected modes of operation. The disconnected mode provides access to key features of the system in clinics that do not have access to the Internet. The Synchronization feature enables disconnected use by providing agency-specific databases to designated computers, known as clinic servers. These servers connect to the Internet each night to upload changes captured throughout the day and to download an updated version of the database to be used the following day. The daily synchronization process involves the following steps:



1. Updates from all clinic servers are submitted until a pre-specified time, usually after peak online production hours. The updates are transmitted to the OLTP database as individual data packets using the same Web methods as in a connected environment. The clinic server essentially provides a store and forward mechanism for the data packets collected throughout the day.
2. After the uploads are completed, the Synchronization (Sync) server generates fresh agency-specific database images to be downloaded. Configuration settings in the Sync server identify which tables of the OLTP database need to be included in the sync process. Stored procedures and the BCP (Bulk Copy Program) utility provide the extraction of data from each table and control whether all data is collected or whether an agency filter is used. The data from each table is compressed and stored in Binary Large Objects (BLOBs) in the Sync database.
3. At a pre-designated time, the clinic server begins the download process. The synchronization process clears the schema on the laptop server before it starts downloading the schema and data. So each download is a clean replacement so that there is no need to log or cache the schema changes. Working from a list of files provided by the Sync server, the clinic server performs the following steps:
    a. Downloads, decompresses, and installs an empty database shell (contains the schema but no data). This shell was loaded in the Sync database as a one time configuration step (not part of nightly process).
    b. Downloads and decompresses data for each required table, then loads that data in the new database shell using the SQL Bulk Insert function.
    c. Downloads, decompresses, and applies scripts to add indexing and perform any other special tasks to the database on the clinic server.

In addition to the data that is downloaded to the clinic server, the database contains a current copy of the application. The application is downloaded to the client automatically at logon from the database based on a comparison of the manifest. This is explained further in section 5, Deployment.

4 Detailed Design

This section details the design of the various architectural components within the application.

The design decisions within the solution are based on the following overall goals for the WIC application:

Scalability

The system is designed for performance with the right layering and separation of concerns. The application is meant to scale linearly against addition of infrastructure resources, which provides for a predictable upgrade path.

Reliability

The solution specifies redundancy and highly-available technology for every mainline component of the production environment. Database clustering, disk mirroring, and stateless components are some of the techniques available in this architecture to ensure reliable, continuous system operation.

Extensibility / Flexibility

This technical architecture permits the straightforward extension of the WIC application through adherence to standards and well-established patterns and practices. The architecture is based on a Services Oriented Architecture (SOA); hence, adding extensions to the systems is easily achieved by adding additional services to the overall solution without having to intrusively change system components.

4.1 User Interface

The user interface design is based on the model view presenter paradigm. The main emphasis of this approach is to separate modeling of domain, presentation, and actions based on user input into three layers:

Model - Manages behavior and data of application domain

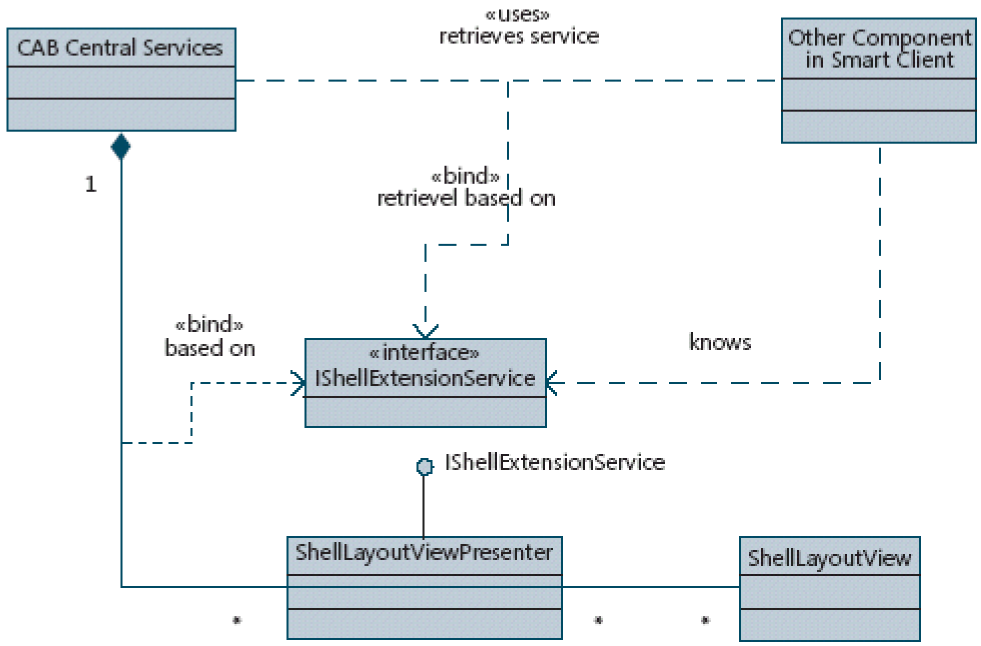
View - Manages display of information

Presenter – Manages visual presentation of the views

In order to further facilitate ease of maintenance, there is a high degree of abstraction used within the development framework to facilitate reuse and enforce a common model for user interface development and integration.

4.1.1 Design Strategy

The typical Model View Presenter (MVP) pattern interaction is detailed in the diagram below.



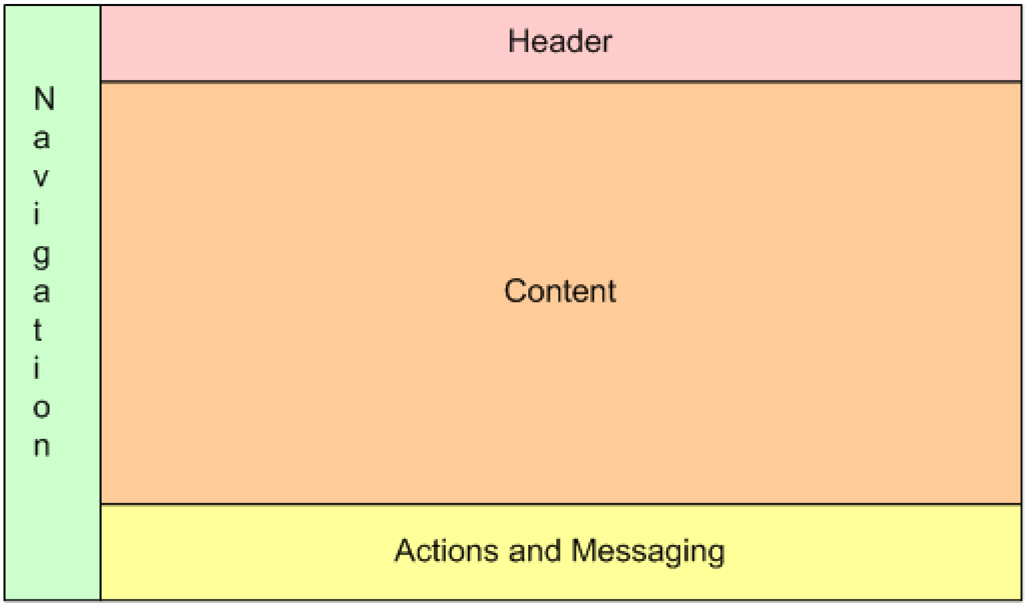
This represents the passive model for an MVP implementation where the data for the user interface is synchronized as a result of an action by the user.

The MVP implementation for this solution is based on the active model for the MVP implementation where the view (screen) is updated automatically via the eventing mechanism through binding events and methods. The binding events are captured by the view itself. However the presenter provides the view with data initially to bind to the controls. The presenter also helps the view to display customized data so that the custom logic is all written in the presenter. Binding takes full control over updating within the view. The active model is used to leverage the benefit of using Microsoft's binding mechanism.

This is achieved by implementing the Observer pattern in the interface that looks for changes in a proactive manner, and hence, leads to a more responsive system. The Observer pattern, which is sometimes called the Publish/Subscribe pattern, is mostly used for dynamic relationships between Objects (mostly views/screens and presenters in our application) where the publisher simply publishes the event (raises the event) based on a change to the state of the object, and the subscribers who have subscribed to the events capture and process the event. Smart Client Software Factory (SCSF) facilitates development by adding the subscribed objects automatically to event publication.

The overall control set is derived mainly from the standard .NET framework controls but has been specialized to adapt to the requirements of the WIC application. The control set contains specialized properties which can be set to monitor changes in the values of other controls without having to programmatically write all the event handlers.

In order to help with usability and ease of training people on the application, the user interaction and visual presentation is layered as shown below.



Individual elements or components of this layout are presented based on the users' permissions and privileges. The specifics of these regions are further detailed in the System Overview DFDD.

4.1.2 Objects and Actions

There are a number of control classes used in the implementation of the user interface. However, this section highlights the ones that are of most significance. For details on others, please refer to the documentation within the code.

Combo box

The WICComboBox is a specialized UI class built on top of the base ComboBox provided by the .NET framework. This specialized combo box handles two different data sources, where one source contains only valid values and the second contains all values, including deactivated ones for historical purposes.

MaskedTextBox

This provides easy application of primary validation based on masks.

Label

The label is specialized to abstract the logic of text and style. With a uniform way of applying these attributes, it is easy to change the display without writing too much code.

Date Time

The DataGridDateTimePicker provides drop-down date functionality in a data grid and is specialized from WICDateTimePicker, which contains the attributes required for displaying different drop-down calendars and easily applying data range checks.

Text Box

The text box is specialized many different ways within the user interface framework. The overarching design decisions for the specialization are:

- Standard application of primary validation.
- Ease of translating data types from the services interfaces.
- Ease of changing the presentation based on attributes of the control.

There is a consistent effort across the interface to standardize control interaction into specific control types to aid maintenance.

Form

There are three basic types of forms in the system:

- *Form without Navigator:* This form provides the same look and feel without navigator control.
- *Form with Navigator:* This form provides the same look and feel with navigator control.
- *Pop-up:* These are the dialogs for messaging and other modeless dialogs that are available as options from within a primary page.

Menu

This set of classes controls the entire navigational framework of the system. The actual navigational paths are stored as a set of XML configuration files, which are then loaded at runtime to draw out the navigation for users based on their security profile.

4.1.3 Interface Design Rules

User interface design is a collection of several different tasks:

- **User interface modeling.** This is the process in which we look at the tasks a program needs to accomplish and decide how to break these tasks into windows and controls.
- **User interface architecture.** This is the logical design we use to divide the functionality of the application into separate objects. Creating a consistent, well-planned design makes it easy to extend, alter, and reuse portions of the user interface framework.
- **User interface coding.** This is the process in which we write the code for managing the user interface with the appropriate classes and objects. We follow the first two steps to lay out a specific user interface model and architecture before we begin this stage.

The user interface modeling has undergone a number of iterations over time to optimize ease of use and user satisfaction. The emphasis of the user interface is on:

- Simplicity.
- Ease of navigation.
- Use of as many standard controls as possible.
- Provision of a user interface that enables users to conduct their work with the minimal number of mistakes.
- Provision of relevant information in accordance with the users’ roles and tasks.

In addition, here are the guidelines for choosing the user interface components:

- Limit the number of characters a text box can accept, and use the key-press event to make sure invalid characters are ignored.
- Use drop-down lists when the user is selecting one of several predefined choices.
- Disable invalid options. This is centralized through the security framework.
- Use radio buttons when choices are mutually exclusive.
- Give feedback on long running tasks using the standard control for wait displays.

These are some of the most important guidelines. There are others that are acquired through years of developing user interfaces. Those guidelines are not detailed here since they deal with aesthetics and the general philosophy in developing user interfaces.

The user interface coding adheres to the overall structure laid out within the user interface framework as described in the System Overview DFDD. The framework is built on a hierarchical model of user interface design with abstracted classes that provide the base functionality for the other classes within the solution. An example of this is the inheritance from the TextBox control. The abstractions are based on usage and common user interface behavior rather than a business function.

4.2 Business

The business rules and logic for the solutions are developed in a separate layer. There are two aspects of the business logic code: the workflow specific to a domain within the application and the rules that govern one or more domains within the application. The actual workflow for the business is implemented within the code and also controlled through the navigation of the user interface. The business rules are implemented as procedural conditions within the code, but wherever possible, these conditions are driven through a collection of system parameters defined in the database. Having the conditional logic configurable through database-persisted parameters allows for quick response to changes in business needs. Further detail on the business classes can be found in Appendix B.

The business services for the solution are implemented in accordance with a Services Oriented Architecture (SOA). There are specific services that need to be invoked by the application installed on the client and are available through WCF services or local invocation as per the needs of the client. As noted in the Architectural Overview, the business services layer is provided through WCF Services. In the classical sense of SOA, these services fall under the category of process-centric services, which are broadly defined as services that encapsulate the specific business logic and processes of an organization.

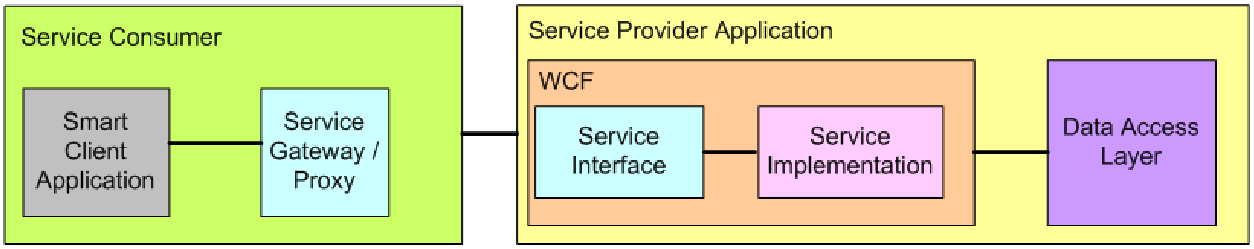
These business services are segregated into the following categories:

- Clinic Services
- Food Management
- Scheduler
- System Administration
- Operation
- Vendor Management
- Finance

Each service has its own interface based on the operation contract. The business service communicates with the client through the proxy that exchanges a business entity. The business entity is prescribed for each unit of work that contains data obtained through the typed dataset in the underlying data access layer. Further information on the management of units of work can be found in the System Overview volume of the Detailed Functional Design Document (DFDD).

4.2.1 Design Strategy

The services are developed as coarse-grained services that expose a collection of methods which are inline with business processes. There are two distinct patterns implemented on the client and server side of the application, depicted in the schematic below.



4.2.2 Objects and Actions

The service implements one 'Getxxxx' and 'Savexxxx' method call for each unit of work. The 'Get' method returns the service entity object that contains the actual data list. The 'Save' method accepts the service entity object and returns the fresh service entity upon successful save. The service is implemented as a WCF service. Each service is implemented by a service contract.

The service interface contains a similar class for a service call, has a specialized class to assemble the information as required by the client, and runs on the server. This allows for the formats in the client and server to change independently from one another.

The HouseholdInterface provides the basic communication to the server and the HouseholdAssembler and takes the values coming from the business layer in the application and converts them into the client format.

4.3 Data Access

The data access layer uses a pure typed dataset that improves the performance and reduces maintenance because of less human-written code. The typed dataset also reduces the usage of stored procedures. The data access layer also houses the business objects and the data contract attribute related to each unit of work and in addition, it houses a convertor and a service entity class. The convertor converts the data in the typed dataset into the corresponding business objects during the 'get' operation and converts the data in the business objects into the datasets. The entity encapsulates the actual data that gets transported between the server and client.

The overall design strategy is based on the following objectives:

- Improve interoperability.
- Reduce the data load by not passing the dataset.

- Provide independent data services that can be deployed and/or hosted individually.

4.4 External Interfaces

This part of the framework deals with all communication with systems or applications outside of the MPSC WIC solution. The implementation of the interfaces is not meant to serve as a means for enterprise integration; rather, it is a loosely coupled part of the overall architecture that solves all issues related to external communication. The core principle behind loose coupling is to reduce the assumptions two parties (components, applications, services, programs, users) make about each other when they exchange information. The more assumptions two parties make about each other and the common protocol, the more efficient the communication can be, but the less tolerant the solution is of interruptions or changes because the parties are tightly coupled to each other.

Overall there are many integration styles that can be used across applications, namely:

- File Transfer
  Have each application produce files of shared data for others to consume and consume files that others have produced.
- Shared Database
  Have the applications store the data they wish to share in a common database.
- Remote Procedure Invocation
  Have each application expose some of its procedures so that they can be invoked remotely, and have applications invoke those to initiate behavior and exchange data.
- Messaging
  Have each application connect to a common messaging system, and exchange data and invoke behavior using messages.

In the case of WIC applications, the interfaces can be developed as file transfer and messaging.

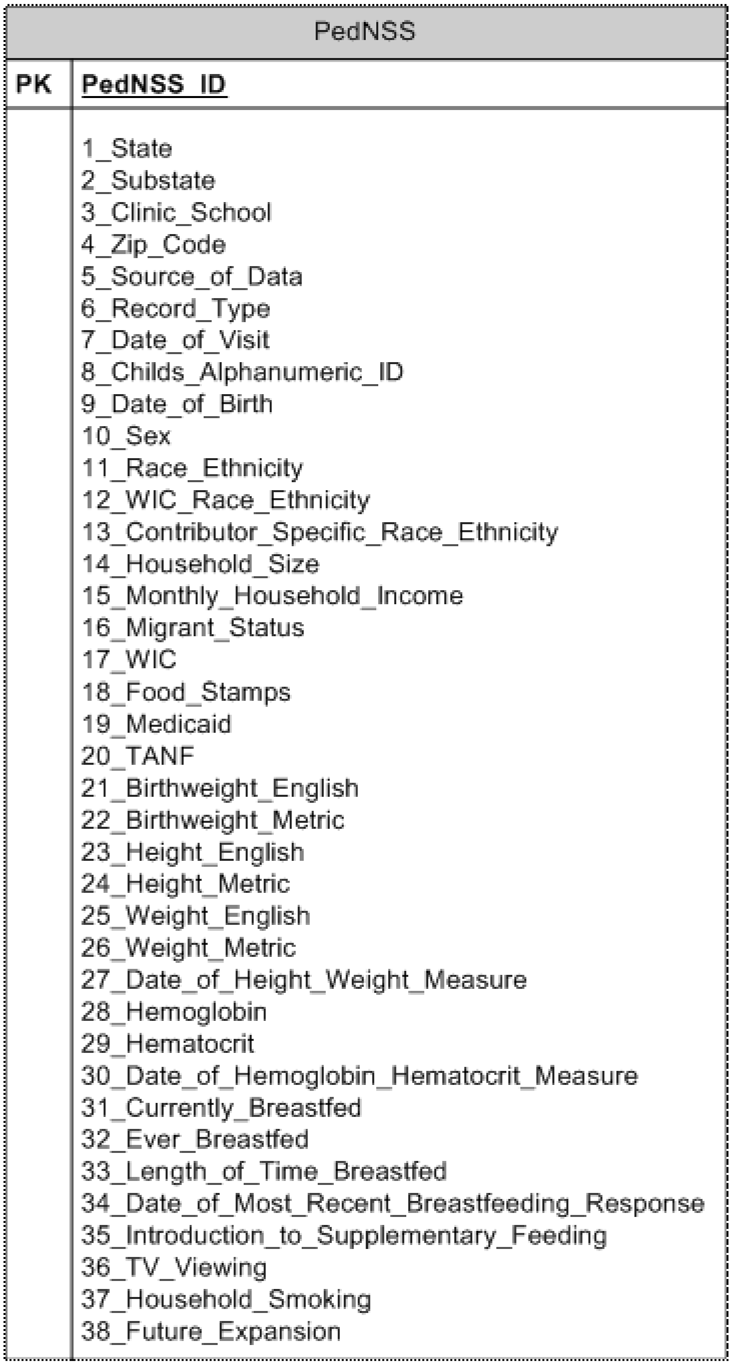
4.4.1 Design Strategy

The interface requirements in the design are driven by the need to provide external systems with data from the WIC system. The invocation of the file transfer is external to the current implementation and needs to be configured based on the target environment. After the extracts have been made available in a flat file, there is a need for manual intervention to make sure that they are transmitted using File Transfer Protocol (FTP).

Every extract or file transfer to external systems is maintained separately and a similar design is followed for the shared database access for other systems. The shared database is not shared for real-time or on-line access; it is shared for information exchange and serves as a one-way feed to external systems, like the EBT host.

4.4.2 Objects and Actions

An example of an extract is the PEDNSS table below. This is an extract from the main database and is subsequently extracted into a flat file using DTS (Data Transformation Service) for FTP transmittal. All extracts are handled in a similar fashion; those that do not require a flat file are available in a separate database.



	PedNSS
PK	**PedNSS_ID**
	1_State
	2_Substate
	3_Clinic_School
	4_Zip_Code
	5_Source_of_Data
	6_Record_Type
	7_Date_of_Visit
	8_Childs_Alphanumeric_ID
	9_Date_of_Birth
	10_Sex
	11_Race_Ethnicity
	12_WIC_Race_Ethnicity
	13_Contributor_Specific_Race_Ethnicity
	14_Household_Size
	15_Monthly_Household_Income
	16_Migrant_Status
	17_WIC
	18_Food_Stamps
	19_Medicaid
	20_TANF
	21_Birthweight_English
	22_Birthweight_Metric
	23_Height_English
	24_Height_Metric
	25_Weight_English
	26_Weight_Metric
	27_Date_of_Height_Weight_Measure
	28_Hemoglobin
	29_Hematocrit
	30_Date_of_Hemoglobin_Hematocrit_Measure
	31_Currently_Breastfed
	32_Ever_Breastfed
	33_Length_of_Time_Breastfed
	34_Date_of_Most_Recent_Breastfeeding_Response
	35_Introduction_to_Supplementary_Feeding
	36_TV_Viewing
	37_Household_Smoking
	38_Future_Expansion

4.5 Reporting

The reporting subsystem provides all the services required to generate reports from the WIC application. The reporting solution is based on Microsoft SQL Server 2005 Reporting Services. The managed reports are created using Report Designer, which facilitates designing a report, previewing the report and publishing the report to the SQL Server 2005 Reporting Services report server. The Report Designer accesses a replicated OLTP database that may also house several denormalized tables. The replicated OLTP is updated nightly after batch processing is completed. The reports can be viewed using the ReportViewer control.

Ad Hoc reports are also created using the Report Designer, but for ad hoc the reporting engine is connected to the Warehouse database. This database is a denormalized database which is queried by end users to enable them to make decisions where there is no managed report. When these tables are populated from the OLTP database, codes and reference tables are expanded (e.g., 'A' will be replaced with 'active', 'T' will be replaced with 'terminated', etc.) and the names of nutrition education classes, risks, foods, etc. are spelled out. Text columns are not brought over because comment columns and other free form text fields are too unpredictable to be useful in an ad hoc query structure.

4.5.1 Design Strategy

The design is based on the following tenets:

- Decouple the reporting engine from the reporting interface.
- Provide ease of implementation for reporting functions from a development and maintenance standpoint.

The interface for the developer is a generic execute type invocation, which allows the developer to specify parameters for the report and the type. For the MPSC solution, the layer that invokes the reporting engine is developed in an open source API for the Report Definition Language (RDL).

4.5.2 Objects and Actions

The components that make up the RDL specific part of the implementation are:

Component	Name	Description
RDL engine	RdlEngine.dll	Provides the reporting engine and rendering services. This is the base engine that other components require.
ReportViewer	ReportViewer	Provides a .Net control for embedding in .Net applications. Displays RDL reports and provides methods for printing and saving to HTML, PDF, and XML.
RDL reader	RdlReader.exe	A MDI application that provides Adobe Reader-like capabilities for RDL reports. This application shows some of the functionality supported by the .Net RDL control.
RDL Designer	RdlDesigner.exe	The WYSIWYG designer allows the user to create standard RDL reports without knowledge of RDL. This includes wizards for creating new reports and for inserting new Tables, Matrixes, and Charts into existing reports. There are property dialogs for all report objects with extensive support for the entire range of power of RDL.

RDL desktop	RdlDesktop.exe	A small desktop report server providing browser access to reports. Point the browser to the URL http://localhost:8080/. Port 8080 is the default and can be modified in the config.xml file. For security, this server only accepts requests from a user's local machine unless reconfigured.
RDL batch	RdlCmd.exe	Batch command executable for creating PDF, XML, HTML files from RDL files.
Data Sources	DataProviders.dll	Provides data access to Web Services, XML, Web logs, CSV files, file directories.

4.6 System Services

4.6.1 Security

Security involves managing risks by providing adequate protections for the confidentiality, privacy, integrity, and availability of information. Security is not a phase in development, design, or deployment but rather a continuous exercise in ensuring that an operating environment is not compromised by a malicious attack. The security solution is audited internally by CIBER's security practice to ensure adherence to best practices.

4.6.1.1 Design Strategy

Most security models for applications can be broken into the following components or principles:

Infrastructure

This deals with securing the physical, data, network, and transport layers within a network. The strategy deployed should ensure that the software and hardware used within an enterprise is not open to attacks.

Application

Application security focuses on securing access to parts of an application and also resources that make up the working application. This area of security deals with the application and presentation tier of the network.

Data

Data security deals with securing access to all information sources that constitute persistent and transient storage, including securing access to data files that might be used to download and upload content to remote sites.

Infrastructure

The infrastructure consists of the network and servers. Included in the network are the DeMilitarized Zone (DMZ), firewalls, and servers. The configuration and security for every part except the servers is based on the hosting solution. The servers are secured (referred to as "hardened" or "hardening" in security parlance). As part of an overall “defense in depth” approach, including multiple layers of security, Microsoft recommends implementing server security measures tailored to the “role” or purpose of each server in the organization. The typical roles covered by Microsoft guidance on security available as **Windows Server 2003 Security Guide** are:

- Domain controllers
- Infrastructure servers
- File servers
- Print servers
- Internet Information Services (IIS) servers
- Internet Authentication Services (IAS) servers
- Certificate Services servers
- Bastion hosts

Based on the hosting solution and environment, CIBER can provide recommendations for the applicable roles in the target environment.

Application

The application software that forms the core of the system is designed and developed with security in mind using appropriate secure application development techniques. In keeping with this philosophy, all new modifications and customizations during the project are developed using industry-accepted secure coding principles.

Controlling user access

- *Authentication:* This is implemented within the solution using a combination of code and database tables. Since there is a tight integration with overall systems behavior, there is no dependence on existing security infrastructure or frameworks. This ensures that the solution utilizes investments in existing technology and infrastructure while providing an extensible framework for security.
- *Authorization:* Most of the authorization flows through the profiles created for authentication. Since the various components in the solution work under an impersonated account, the management of roles involved in the authority over various resources (like database and file systems) is minimized.
- *Role-based access:* Roles are heavily used across the solution to provide the administrator with manageable groups of permission sets. Roles are used to determine field availability and actions (menu items) on the user interface. The application access control facility is highly flexible, allowing administrators to assign whatever privileges are needed by users to perform their job function. This allows fine-grained control of access privileges and permits the implementation of a role-based “least privilege” model, widely recognized as the best application security model because it avoids assignment of excess privileges such as might be the case in less granular application security schemes.
- *Session management:* The user interface is Windows forms-based and it accesses services on a Web server, in case of connected solutions, or synchronization services, in case of a disconnected solution. It is

imperative that users not monopolize resources on the Web server; otherwise, it affects scalability adversely. Additionally, unlimited access to Web server resources is a great security risk, as it provides an opportunity to launch a 'brute force' attack on the system. In order to address these issues, we make use of the .NET-supplied infrastructure for session management.

- *Navigation:* Navigation within a system determines usability and functionality, which has an effect on security. Limiting the amount of functionality available to a user automatically limits the surface area of attack in an application; in addition, it provides information security because the users can only see what they are authorized to access. Developing specific navigational paths and views for specific user roles within the system enforces these principles within the current solution.
- *Auditing and logging:* The solution utilizes the logging features available in Enterprise Library and Microsoft SQL Server to provide robust logging of all types of access to the system, from security information to changes made to database fields.
- *Screen elements / Masking of data:* While navigation limits the areas of the system that can be accessed, the specific pieces of information presented still need to be protected/unprotected (hidden/shown) based on access privileges. These types of screen elements are mapped back to user roles.

<u>Maintaining user access</u>

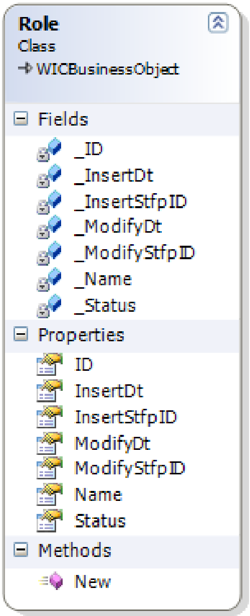
The user access, roles, and permissions are an integral part of the solution. All these are maintained using screens provided in the current application, which allow for very granular controls on the actions that can be performed by the user within the system.

Data

This uses the facilities provided in the Microsoft SQL Server. All access to the system is based on using a common user profile for all database operations since this promotes the most effective use of connection pooling, and hence, provides effective scalability to the solution. This user profile is used by system components within the Data Access Layer and is never exposed to the client. All client calls are passed to the Data Access Layer for data persistence and management, making this a very secure and robust solution.

4.6.1.2 Objects and Actions

There are a number of classes used to control the access and permissions to on screen data as shown below.

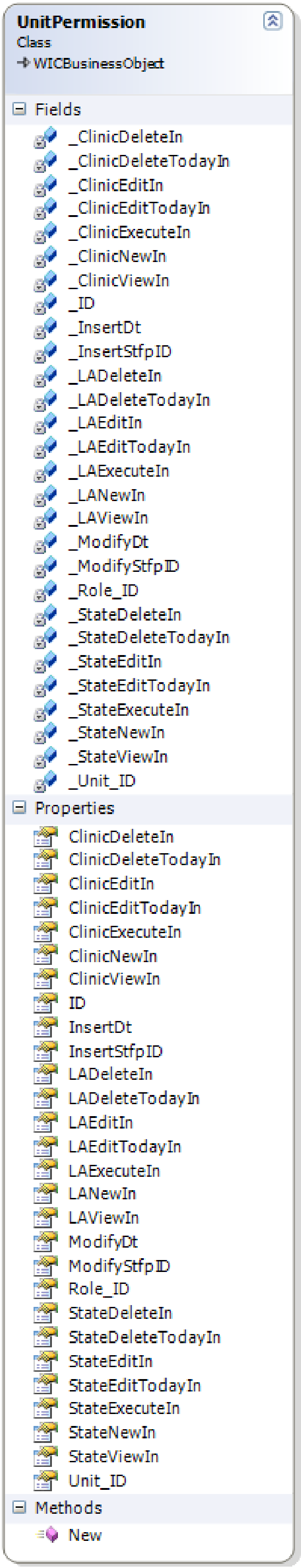


Role
Class
→ WICBusinessObject

Fields
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _Status

Properties
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- Status

Methods
- New

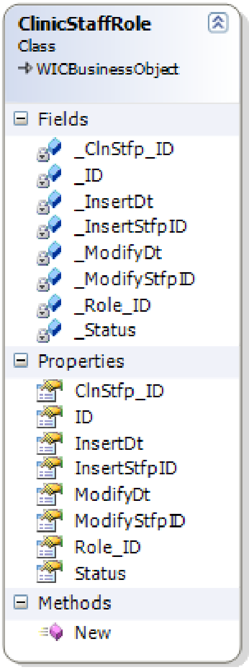


UnitPermission
Class
→ WICBusinessObject

Fields
- _ClinicDeleteIn
- _ClinicDeleteTodayIn
- _ClinicEditIn
- _ClinicEditTodayIn
- _ClinicExecuteIn
- _ClinicNewIn
- _ClinicViewIn
- _ID
- _InsertDt
- _InsertStfpID
- _LADeleteIn
- _LADeleteTodayIn
- _LAEditIn
- _LAEditTodayIn
- _LAExecuteIn
- _LANewIn
- _LAViewIn
- _ModifyDt
- _ModifyStfpID
- _Role_ID
- _StateDeleteIn
- _StateDeleteTodayIn
- _StateEditIn
- _StateEditTodayIn
- _StateExecuteIn
- _StateNewIn
- _StateViewIn
- _Unit_ID

Properties
- ClinicDeleteIn
- ClinicDeleteTodayIn
- ClinicEditIn
- ClinicEditTodayIn
- ClinicExecuteIn
- ClinicNewIn
- ClinicViewIn
- ID
- InsertDt
- InsertStfpID
- LADeleteIn
- LADeleteTodayIn
- LAEditIn
- LAEditTodayIn
- LAExecuteIn
- LANewIn
- LAViewIn
- ModifyDt
- ModifyStfpID
- Role_ID
- StateDeleteIn
- StateDeleteTodayIn
- StateEditIn
- StateEditTodayIn
- StateExecuteIn
- StateNewIn
- StateViewIn
- Unit_ID

Methods
- New



ClinicStaffRole
Class
→ WICBusinessObject

Fields
- _ClnStfp_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Role_ID
- _Status

Properties
- ClnStfp_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Role_ID
- Status

Methods
- New

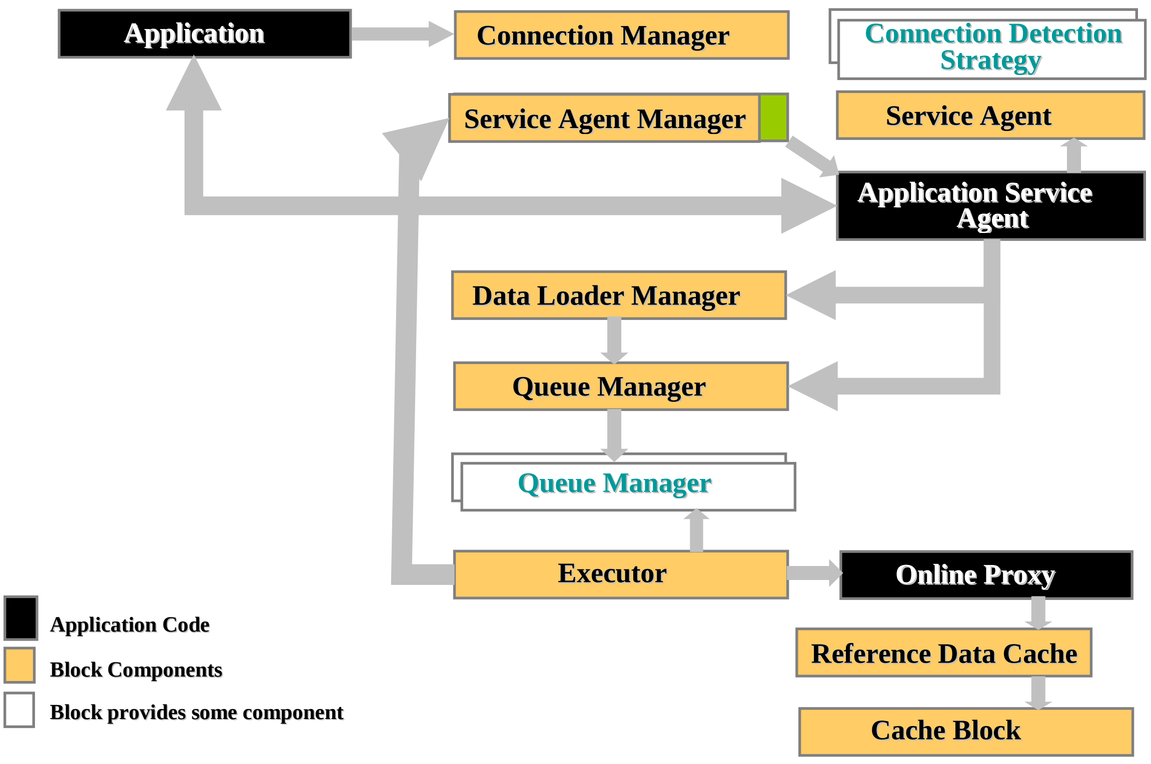
The Security Controller manages the specific data packets based on agency, clinic, and role. Since this is a profile that is stored against a login for a user, it is applied uniformly to the entire set of user interface screens. The methods on the class also supply the functionality for checking for actions against the data without having to retrieve the security permissions each time.

4.6.2 Cache

One of the best ways for reducing the amount of server roundtrips is to store the data locally on the client. This is one of the primary goals and advantages of the Smart Client architecture. The cache management for this solution focuses on the client-side caching for information. The caching is done using Enterprise Library's Caching Application Block. The block caches the most-used data such as codes, system parameters, error messages, local agencies, and clinic lists in isolated storage. The isolate storage is created per MPSC system for each user. The data in the cache are serialized.

4.6.2.1 Design Strategy

The design is based on the Smart Client Offline Application Block provided by Microsoft. The overall architecture is described in the following schematic.



The Offline Application Block provides the basic functions required by Smart Client applications with offline capabilities. Its essential features include:

- Detecting the presence or absence of network connectivity.
- Notifying all registered components when the connection state changes.
- Downloading and caching the reference data that allows the application to function when the network connection is not available.

- Storing message data locally while the application is offline.
- Synchronizing message data with the server when the network connection becomes available.

4.6.2.2 Objects and Actions

Most of the objects are defined within the application block, and there are additions made specific to the solution. The table below describes the basic components in the solution.

Elements	**Description**
Connection Detection Strategy	Detects the current state of the physical connection.
ConnectionManager	Manages connection state services related to physical network access. Uses pluggable connection detection components to determine the connection state.
Executor	When online, takes messages off the queue and calls the Online Proxy responsible for sending them to the remote service. Additionally, sends responses from the remote service back to the application.
Queue Storage	Provides the data store used to hold message data and operations to be delivered to the server when the application is again online. For this solution, the storage is a mix of memory, database, and Microsoft Message Queues (MSMQ).
QueueManager	Behaves as a façade for the Queue Storage Providers. It provides methods for enqueuing and dequeuing the messages. This provides the serialization required within the solution.
DataLoaderManager	Provides a facility to allow the application to request reference data to be downloaded at an appropriate time for use during operations. This works in an asynchronous manner to download the reference data.

ReferenceDataCache	Behaves as a façade for the Reference Data Management subsystem. Responsible for ensuring that data required by the application is stored and can be accessed locally (cached) while the application is offline.
Cache Block	Provides an implementation of physical cache operations.
Application Service Agent	Provides the ability to queue messages. It also provides a channel for getting results back to the application.
Online Proxy	Class created by the application developer that has the responsibility of communicating with the remote service that provides the business capability. The Online Proxy also has the responsibility of storing the reference data in the cache, if required.
ServiceAgent	Provides the base class implemented by all application-supplied Service Agents. The Service Agent base class is responsible for registering the service agent with the service agent registry.
ServiceAgentManager	The Service Agent Manager returns the results back to the appropriate Service Agents after processing.

The connection strategy for the solution is to build for detection of connectivity and to allow for the speed of the network, as far as possible, by introducing delays between communications retries. This helps in ensuring that the user experience is maintained through slow connections.

The queue manager is also a mechanism used for synchronization of offline content with the server and provides a robust mechanism for transferring data between client and server.

4.6.3 Exception Handling

An exception occurs when a predefined assumption within the application is broken. The exception management framework for the solution provides the basic building blocks to:

- Detect exception.
- Perform code clean up.
- Wrap one exception inside another.
- Replace one exception with another.
- Log and report error information.

- Generate events that can be monitored externally to assist system operation.

The exception management framework for MPSC provides the following functions:

- Allows the solution to gracefully handle unforeseen conditions.
- Acts as a debugging aid to efficiently determine the root cause of the exception.

4.6.3.1 Design Strategy

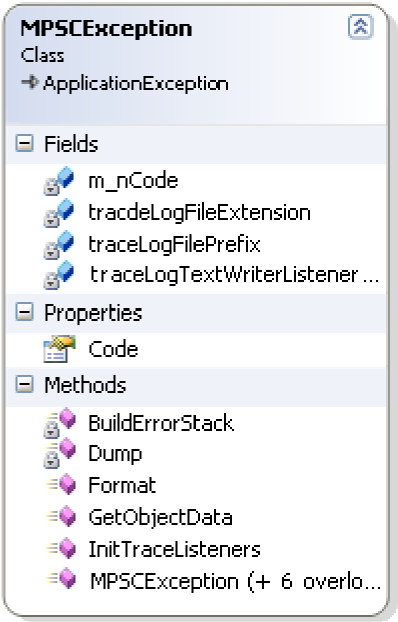
The design is based on developing a customer exception handler that wraps all the logic of exception propagation and logging within itself. The design strategy, which is also a coding practice, is built around the following guidelines:

- Programmatically check for any exception errors such as NullReference or ArgumentZero exceptions. Use the exception handler to catch any exception that comes from entities outside the boundary, such as WCF services, external interfaces, etc. The application should also catch an exception that results from an unhandled condition from these sources.
- Try to recover from the exception. If possible the application should itself try to take a different path to obtain resolution.
- Add the contextual information to the exception.
- Propagate the exception to the caller.

The actual information management and logging of the exception is handled within a generic exception class that utilizes Microsoft’s Enterprise Instrumentation Framework to write the information to the system’s database. All exceptions are recorded and viewable through the System Administration user interface.

4.6.3.2 Objects and Actions

The diagram below shows the main class that wraps the overall logic for handling and logging exceptions. The MPSC exception class inherits fields, properties, and methods from the Microsoft class in addition to the MPSC-specific ones. In addition to this class, there are coding guidelines that help in the implementation of exception handling throughout the solution. The basic interface to use this class is the overloaded constructor. In addition, the BuildErrorStack and Dump methods provide ways to add valuable information to the exception and then write it out to a configurable location.



MPSCException
Class
→ApplicationException

⊟ Fields
- m_nCode
- tracdeLogFileExtension
- traceLogFilePrefix
- traceLogTextWriterListener ...

⊟ Properties
- Code

⊟ Methods
- BuildErrorStack
- Dump
- Format
- GetObjectData
- InitTraceListeners
- MPSCException (+ 6 overlo...

4.7 Database

The database is modeled based on real business entities, as far as possible, and is used to drive the object design of the application. The Entity Relationship Diagram (ERD) is attached in the Appendix.

The MPSC system requires five separate databases. Separation into these areas is standard because it allows the separation of the processing load across multiple servers. Servers are usually configured differently based on the processes that they will run. Online servers may have faster processors and more memory, whereas reporting and warehouse servers will have more disk space and faster i/o channels.

4.7.1 OLTP

OLTP stands for On-Line Transactional Processing. This database contains the tables to which the online system reads and writes. These are the transactional tables that are used by the functional areas every day.

4.7.2 Audit

There is an exact copy of each online table schema in a separate audit database. These tables are populated by the execution of triggers on the online tables. Inserts, updates, and deletes trigger writes to the audit database. Database Administrators will use MS SQL tools to access the audit database to research data issues or concerns. There is no user interface to access the audit database. There is no referential integrity in the audit database.

4.7.3 Warehouse

The warehouse database contains denormalized versions of the data required for ad hoc reporting. The stored procedures that are used to populate the warehouse tables are in the warehouse database. There are scheduled jobs that populate the warehouse tables. The following Warehouse tables are included to address ad hoc reporting needs:

Clinics

Complaints

EBT Card Transactions

EBT Issuances

EBT Redemptions

FIs

Inventory – Non Serialized

Inventory – Serialized

Issued Containers

Nutrition Education Pamphlets

Nutrition Education Topics, Sub-Topics, & Counseling Points

Participants

Participant Snapshot

Participation

Referrals

Returned Formula

Risk Data

Scheduler Appointments

Scheduler Nutrition Education Classes

Scheduler Slots

Staff Competency

Staff Credential

Staff State Training

Surveys

Time Study

Vendor Civil Money Penalties

Vendor Details

Vendor High risk

Vendor Training

Vendor Visit History

4.7.4 Reports

The reports database contains the OLTP replicated tables that are populated by nightly batch jobs. In addition this database also contains denormalized tables that present aggregated data in the required reporting formats. The database may also house the stored procedures that are called from the report designer. If runtime requirements are introduced for production reports that require a level of performance not previously specified and that are an exception to how most production reports are managed, report tables are created and populated on a scheduled basis. The reports are requested through reporting services asynchronously so that users can move on the other tasks after placing the request for the report.

4.7.5 Synchronization

The synchronization database contains the tables necessary to support the data needs of the disconnected clinic servers. Synchronization is described in more detail in section 3.8 Disconnected Operations.

5 System Deployment

The deployment of the system is highly scripted. Scripts have been developed to deploy the solution internally within the development environment. The scripts provide the necessary automation to ensure that the system can be rebuilt with minimal effort.

The deployment methodology is tightly coupled with the software development life cycle and is delivered through "continuous integration." Continuous integration is the process of starting a build whenever code is checked into the source control server or at a predetermined interval. This has the following advantages:

- Build breaks are caught early.
- It assures the developer that the build on top of which he is rebuilding is the latest one and would cause less regression.
- The feedback loop is smaller. A developer does not have to wait for the end of the day or week to find out how his check-in affected the build.
- Integration testing moves up in the chain. Every check-in goes through the integration testing where issues are caught early on.
- Better development processes are enforced with more accountability on each developer.

The entire automation is controlled through scripts developed for various parts of the deployment. The whole build and deployment process is built based on the MSBuild process. The whole build process is handled through a single process that includes building the application, deploying database changes, and hosting the services.

5.1 Server

The server deployment assumes that the required systems software and patches, as mandated by Microsoft, have been applied to the servers.

There are two distinct server-side installations:

- *Web Server:* The Web Server installation consists of all the settings required for deploying a WCF service and the configurations that accompany that. Some of the configurations considered are:
    - Entries into the services discovery.
    - Setup required for the virtual web sites within IIS.
    - Replacement within web.config.
- *Database Server:* The database server has the OLTP and the reporting databases. The deployment for these are scripted using T-SQL and are wrapped in the same open source tool suite mentioned above.

5.2 Client

The client update process is based on the Application Updater component authored by Microsoft. The application block has been enhanced and changed in places to suit the specific needs of the MPSC solution.

The overall design goals for the client updating process are:

- Update the client on demand without any manual intervention.
- Update only the parts that have been changed.
- Employ an extensible packaging process in order to add or remove components with ease.
- Provide a secure deployment which ensures integrity of the packages being installed on the client.

The specific steps involved in communication between the client and the server are described below. Within the context of the MPSC solution, this component is referred to as the "AppUpdater" and works equally efficiently for connected and disconnected clinics.

5.2.1 Application Publishing Process

1. Application files are streamed, converted to Byte Arrays, and saved as BLOBs in the OLTP database
2. A nightly synchronization process copies application BLOBs to clinic servers so clients in disconnected clinics can retrieve the latest version of the application.

5.2.2 Initial Application Install Process

1. The user navigates to the "Click-For-WIC" Web page.
2. A bootstrap utility (WICDownloader) is downloaded to the desktop.
3. WICDownloader downloads AppStart and AppUpdater from the Web service and creates a "WIC" shortcut on the desktop which references AppStart.

5.2.3 Daily Application Update Process

1. The user double-clicks the WIC icon on the desktop, initializing AppStart.
2. AppStart starts the current version of AppUpdater when the WIC icon is clicked.
3. AppUpdater obtains a new manifest through the Web service, compares it with its own, and if an update is available, begins downloading files. The manifest file contains information about all application files and folder structures on the client.
    a) The scavenging feature reduces network load and time required for the download by downloading only new or modified files. Files that have not changed from the previous version are simply copied to the new version folder.
    b) Cleanup tasks are performed (manifest replaced, temp files and old versions deleted - keeps last 3 versions only).
4. The newest version of the WIC application is started and the user is prompted to logon.

6 Systemic Qualities

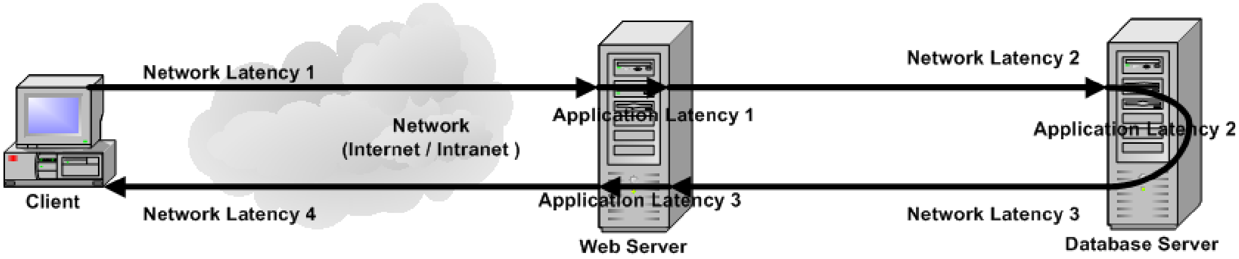
There are a number of areas within the architecture that need to be considered for the overall solution and applied system-wide instead of within the scope of a specific component. Outlined in this section are the most important considerations for system-wide architectural requirements.

6.1 Performance

Performance of a solution is its ability to respond under various load conditions. This is usually quantified as:

- *Throughput:* Throughput is the number of requests that an application can serve in a specified unit of time. Throughput is typically specified in requests per second.
- *Availability:* Availability is the percentage of time an application is responsive to client requests.
- *Scalability:* Scalability is discussed in detail in the next section.
- *Response Time:* The response time is a measure of time the user has to wait in order to complete a request. This usually is the basis of user perception for performance of all components of the solution.

The typical latencies with a Web-based solution are detailed below.



The whole emphasis of a performance approach is to minimize latencies, which keeps the aforementioned factors of throughput, availability, scalability, and response time within acceptable limits.

The requirements for the solution stipulate that the application needs to be able to provide information back to operators in less than five seconds; target response time is less than three seconds from panel to panel within the application. Smart client is designed to take a bit longer to initially download data and to save data, but this has a payback in multiples as panel-to-panel navigation and data updating within the application approaches sub-second response.

The primary consideration with performance requirements is the speed of the network. The architectural solution is to limit the amount of data being returned to the user for any particular request. Composite and sub-view patterns are employed to limit the amount of data required to be transported over the wire. The response packets are further compressed to ensure optimal usage of the network bandwidth.

From a persistence view, caching strategies are employed to minimize database access for commonly used data. The client downloads this data and caches it locally instead of having to request it with every packet of information.

The infrastructure available across the consortium states determines a lot about response times, and we continue to ensure that the overall solution performance is acceptable.

6.2 Scalability

Scalability is the ability of an application to maintain or improve performance as the user load increases. Scalability also refers to the ability of an application to recognize performance benefits as server resources increase. This system-wide concern primarily references the server since the client is not a bottleneck in a Smart Client system.

The design of the application is loosely coupled with each of the following layers of the solution candidates for being scaled horizontally (adding more hardware):

- Web Server
- Database Server

The solution based on a smart-client model is meant to scale to take advantage of all the hardware (especially server) resources provided within these resource areas.

The degree of scaling to support the system demands will vary from state-to-state. CIBER will work with each state to determine the proper server configurations based on caseload and number of concurrent users.

6.3 Usability

Usability is a quality attribute that assesses how easy user interfaces are to use. It also encompasses how easy a system is to learn and memorize and how efficient it is to navigate.

The system is built on the following basic design principles that help satisfy the aforementioned requirements:

- The regions within the user interface are layered with specific functional attributes in mind. Please refer to section 4.1 for more details.
- The user interface deploys an easy navigation framework to ensure accessibility to all parts of the application without overwhelming the user with too much to do.
- The layout for the user interface combines information required for a business function within a set of screens.
- There is online user help to assist users with common tasks.
- The scheduler closely resembles a calendar and Microsoft Outlook type interface for ease of use.
- There is a quick search available from various parts of the system.
- The validation framework within the solution provides intuitive messages to help users.

7 Future Considerations

Any architecture process cannot be complete without acknowledging the fact that it needs to adapt to technological change. This section details the changes in technologies that can impact the solution. This is by no means an exhaustive review of all the technology changes with the solution set.

In addition to changes in technology, any WIC system needs to be able to adapt to upcoming changes in the governance or rules of the WIC program. The current architecture easily supports these changes because it isolates a number of the rules as parameters. However, if there are material changes to the overall rules and governance of the WIC program at the State or Federal level, it might necessitate a review of the codebase in the solution at a more detailed level.

Technology

Microsoft is rapidly adding to the current software base for Smart Client applications. The current Release To Manufacturing (RTM) release for the .NET Framework is 2.0. However, .NET Framework 3.0 is already in beta. This latest version of the framework is not being used for this project because of the late release date and the stabilization period required by any new technology. In addition, there is the possibility of a new server operating system being released in the next year. Since the system is not being developed with specific dependencies on the facilities provided by the server components (except the .NET Framework), there should be few to no problems expected with upgrading to the latest release of the operating system.

- *Windows Presentation Foundation*

  Windows Presentation Foundation (formerly code named "Avalon") is Microsoft's unified presentation subsystem for Windows and is exposed through .NET Framework v3.0, Windows Vista's managed-code programming model.

  Windows Presentation Foundation (WPF) consists of a display engine that takes full advantage of modern graphics hardware and an extensible set of managed classes that development teams can use to create rich, visually stunning applications. WPF also introduces Extensible Application Markup Language (XAML), which enables developers and designers to use an XML-based model to declaratively specify the desired user interface (UI) behavior.

  Windows Presentation Foundation provides a unified approach to the user interface, 2D and 3D graphics, animation, documents and media. It allows designers to be an integral part of the application development process.

  Due to the change in the basic systems infrastructure provided by Microsoft, this changes the way some of the interfaces of the future will be written and thought out. However, to retrofit the current implementation to the WPF requires a lot of rework since WPF is a completely different base architecture for user interface development than the one used for the MPSC WIC application.

- *LINQ*

  The next big challenge in programming technology is to reduce the complexity of accessing and integrating information that is not natively defined using Object-Oriented (OO) technology. The two most common sources of non-OO information are relational databases and XML.

  Rather than add relational or XML-specific features to our programming languages and runtime, we have taken a more general approach and have added general purpose query facilities to the .NET framework that apply to all sources of information, not just relational or XML data. This facility is called .NET Language Integrated Query (LINQ).

  Language Integrated Query allows query expressions to benefit from the rich metadata, compile-time syntax checking, static typing and IntelliSense previously available only to imperative code. Language Integrated Query also allows a single general purpose declarative query facility to be applied to all in-memory information, not just information from external sources.

  This is a facility of the future and requires some significant changes to the base infrastructure of the data retrieval for complete adoption. There is a possibility of refactoring the code to integrate this technology, but it would be a significant change moving forward.

8 Dependencies

8.1 Tools

Tiers can be defined at a number of levels: conceptual, logical and physical.

At a *conceptual* level, they represent distinct and cohesive aggregations of functionality. Typical tiers include:

- *Client,* representing the point at which data is consumed by the system's users.
- *Presentation,* supporting generation and customization of content for specific client device types, languages, or based on other personalization parameters.
- *Business logic,* centralizing and encapsulating the business logic of the system and aggregating disparate backend services.
- *Integration,* wrapping access to backend resources in higher level abstractions.
- *Resource,* where databases, devices, and legacy systems reside.

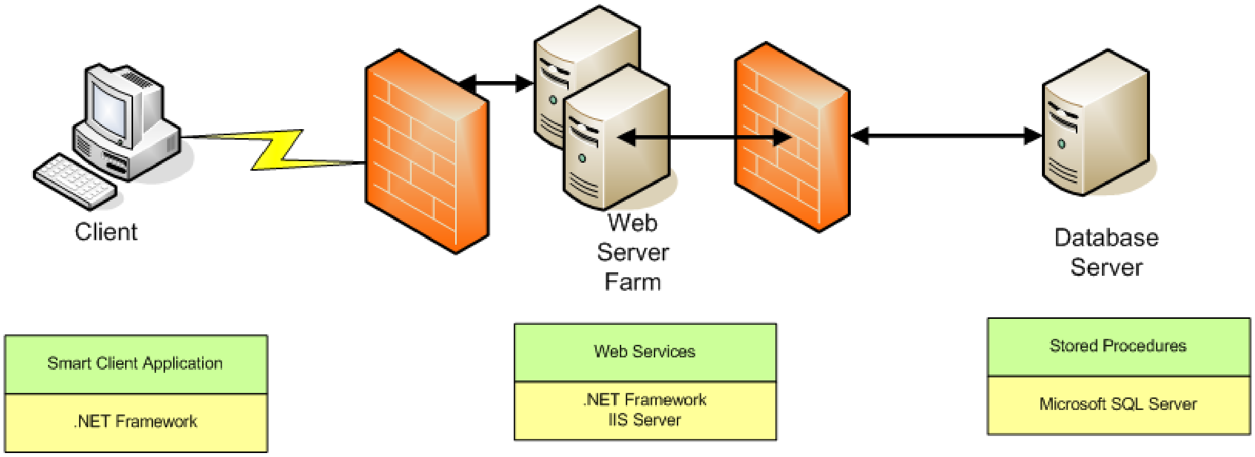
Product	Version	Description	Tier
COTS Software			
Windows 2003 Server	Enterprise	Base operating system	All
Microsoft SQL Server	2005	A relational datastore for managing enterprise information	Resource
Microsoft SQL Server Reporting Services	2005	Enterprise reporting tool	Resource
Software Frameworks			
Enterprise Library	3.1	Library of application blocks	All
Tools			
Microsoft Visual Studio	2005	Integrated development environment	All
Microsoft Visio	2003	A UML design tool	All
StarTeam	2005	A full feature source control and configuration management tool	All (Development, Integration, Test environments at CIBER Harrisburg)

Product	Version	Description	Tier
QualityCenter	8.0	A test management tool to track defects, create and manage test cases, and automate regression testing.	All (Test environments at CIBER Harrisburg)

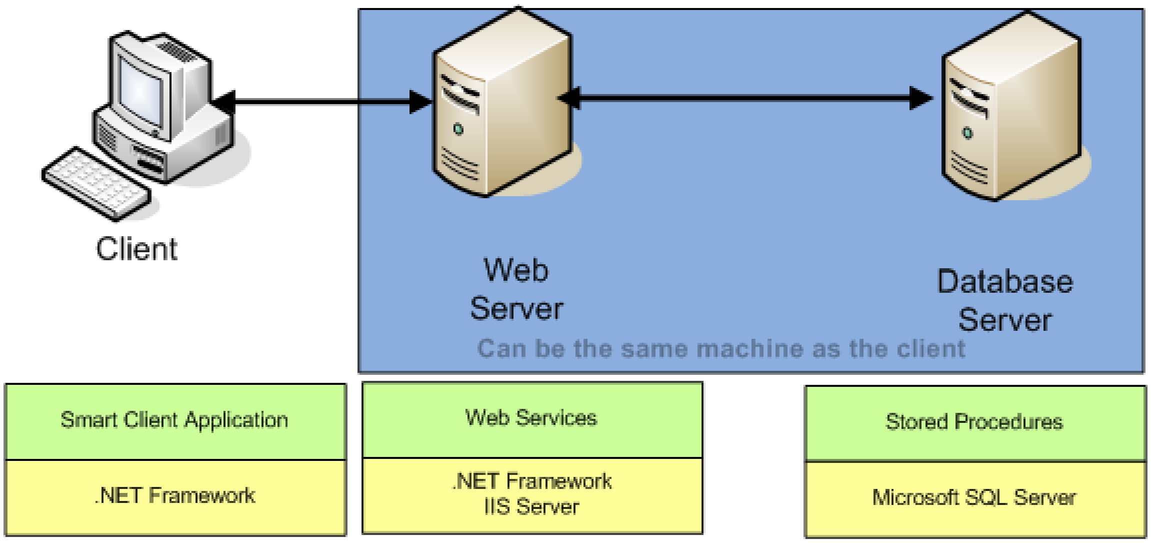
8.2 Technologies

The overall solution for MPSC is based around Microsoft technologies and built on the .NET Framework. The schematics below depict the overall deployment of the system.

The following schematic depicts a connected operation.



This schematic depicts a disconnected operation.



8.2.1 Smart Client Application

This component is installed on each client machine that needs access to the proposed MPSC WIC system. The installation is a seamless process handled automatically by the application. The application runs in two modes:

- Connected
- Disconnected

The packaging from CIBER provides all the dependent application files for the application with the exception of the .NET Framework. The system requires .NET Framework 3.0 for its execution

8.2.2 Database Server

The OLTP database server stored procedures are accessed from the web server using ADO.NET. All the stored procedures are written in T-SQL; hence, there is dependence on Microsoft SQL Server 2005 or later.

The database server does not need the .NET framework since there are no processes executed from within the database.

8.2.3 Web Server

This acts as the "endpoint" for all calls made for database interaction with the system. This is used by the online application as well as the synchronization mechanism for off-line database synchronization (not depicted in the schematic for the sake of simplicity). The "endpoint" is both an implementation and standard used to describe WCF services communications. The implementation at this time is based on the Microsoft standard and tested against the libraries provided in the .NET framework for IIS. The web server is installed with IIS 6.0 which can handle 4000 concurrent HTTP connections operated by Windows Server 2003. In order to prevent malicious attacks it is preferred to set the connection limit. For example if the web server needs to serve 50 clients simultaneously then it would be suggested that the number of connections is set to 200 with connection timeout to 120 seconds. In addition each WCF Service hosted in IIS has its own configuration data such as <ServiceThrottling> which is set according to the number of requests it should process simultaneously. The default is 10.

8.3 Infrastructure

The infrastructure requirements for the application cover the following:

- Servers
- Network
- Client configurations
- Bandwidth

In addition, there are a number of environments required for the development and deployment of the solution. The next section addresses the environments required by the consortium states to test and deploy the application.

8.3.1 Environments

There are two environments required by the consortium states:

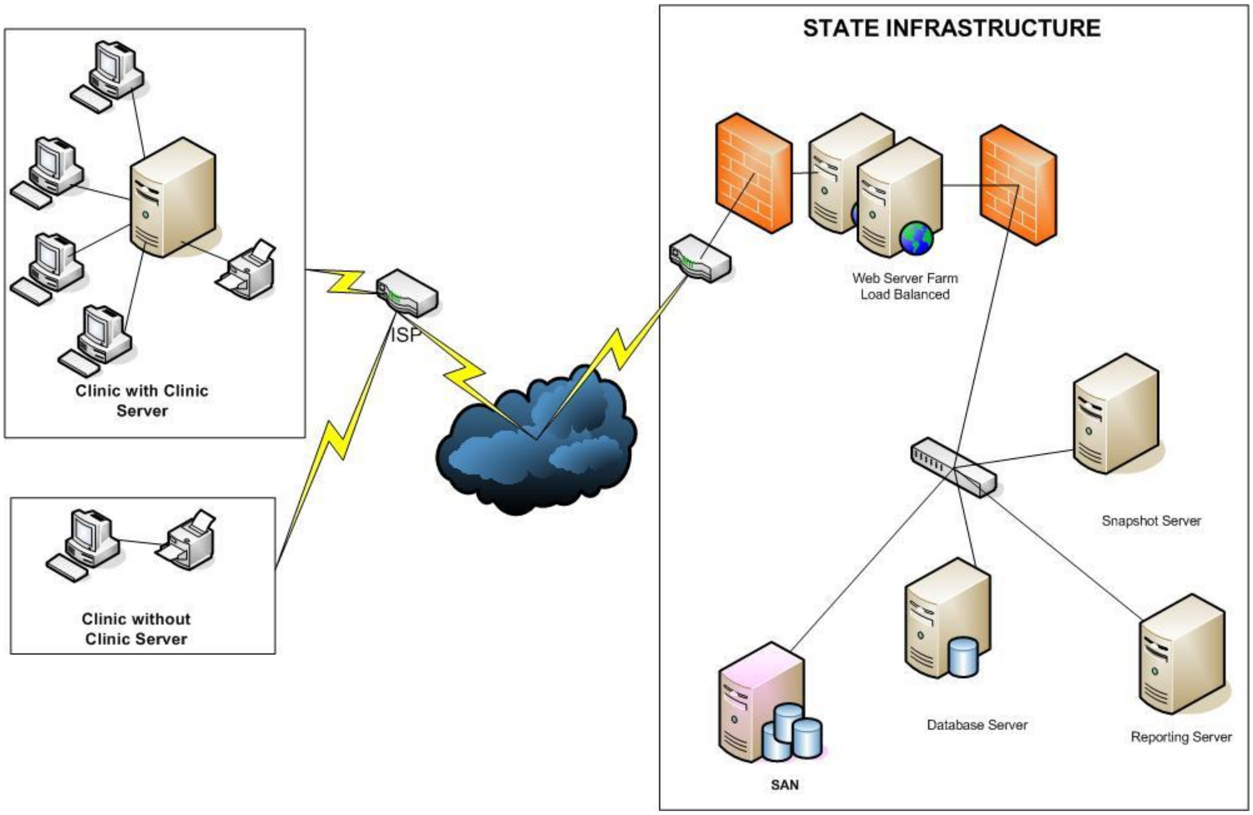
Test / UAT

This is the test environment for the application and also hosts UAT. Although this is one environment, it can have multiple installations of the application to support the various versions required for testing and deployment of the solution. This also may act as the back-up for the production environment in order to provide disaster recovery.

Production

The production environment provides a redundant infrastructure that provides a high degree of availability.

The schematic below details the overall infrastructure required for the MPSC solution. The two environments mentioned above are a mirror of each other, which provides basic disaster recovery through redundant environments. Wherever possible, these environments are also hosted in separate physical locations to provide disaster recovery against any environmental reasons.



The functions performed by the various servers are as follows:

- **Disconnected clinic server**
  This serves as the local server for a set of client computers. Its serves the functions of Web, application and database in one, while providing the synchronization functions with the main stateside servers.
- **Web Server**
  This hosts all the WCF services utilized for the solution. This is the main connection point for all the communication with the stateside host environment. Each business services has its own WCF services hosted in IIS.
- **Database Server**
  This is persistent store within the solution and contains all On Line Transaction Processing instances for the solution. There is one instance of the database that supports the entire application.
- **Snapshot Server**
  This serves as the main resource for all synchronization with disconnected operations. This also has a copy of the OLTP database, which is used to create sets of data to be synchronized with disconnected clients.

- **Data Warehouse Server**
  The data warehouse will serve as the base for data acquisition for all ad hoc reporting needs within the system.

8.3.2 Hardware

The recommendations in this section represent the best practice scenarios as recommended for the consortium. However, there might be specific situations that apply to individual states that might need to be addressed on a case by case basis.

The following are not covered in these specifications:

- Specific recommendations around types of disk fail-over systems, like RAID 0 or RAID 5, since it is assumed that the storage will be on a Storage Area Network (SAN) or Network Area Storage (NAS).
- Intel processors are used for specifications; equivalent AMD chipsets can also be used.
- There are no specifications on the type of network for connectivity between the sites. This is dependent on the specific needs of the individual clinics as described in the Site Survey Results.

Client Configuration

Connected

- Processor – Pentium 4 3GHz or greater
- RAM – 512 MB (1024 recommended)
- Storage – 10 GB hard drive
- Operating System – Windows XP Home SP2 or Professional SP2

Disconnected

- Processor – Pentium 4 3GHz or greater
- RAM – 1024 MB (2048 MB recommended)
- Storage – 20 GB hard drive (40 GB hard drive recommended)
- Operating System – Windows XP Home SP2 or Professional SP2

Server Configuration

Web Server

- Processor – Intel Dual Core Xeon 2.33 GHz or greater
- RAM – 4 GB
- Storage – 20 GB
- Operating System – Windows Server 2003 Enterprise R2
- Software - .Net Framework 3.0

Database Server

- Processor – Intel Dual Core Xeon 2.33 GHz or greater

- RAM – 16 GB
- Storage – 100 GB
- Operating System – Windows Server 2003 Datacenter R2

Snapshot Server

- Processor – Intel Dual Core Xeon 2.33 GHz or greater
- RAM – 8 GB
- Storage – 60 GB
- Operating System – Windows Server 2003 Datacenter R2

Data Warehouse Server

- Processor – Intel Dual Core Xeon 2.33 GHz or greater
- RAM – 4 GB
- Storage – 100 GB
- Operating System – Windows Server 2003 Datacenter R2

Disconnected Clinic Server
(These specification will vary depending on the size of the clinic, the number of computers that connect to the server, and also the number of participants serviced.)

- Processor – Intel Dual Core Xeon 2.33 GHz or greater
- RAM – 8 GB
- Storage – 60 GB
- Operating System – Windows XP Professional SP2 or Windows Server 2003 Enterprise R2

Firewall, Router and other Devices

There are no application dependencies with respect to network access and control other than they be secure. These security considerations must be consistent with the standards of the hosting location as well. CIBER will make recommendations in conjunction with reviewing the infrastructure and hosting environment of each state.

8.3.3 Network

Client

Bandwidth

Connected

# of Workstations in Clinic	Bandwidth
1	256 Kbps

# of Workstations in Clinic	Bandwidth
2-5	750 Kbps
6 or more	1500 Kbps

Disconnected

# of Computers Concurrently Synchronizing at Agency / Clinic	Bandwidth
1-2	750 Kbps
3-5	1500 Kbps

Server

Ports

Web Server

- External – 443
- Internal - 443

Database Server
Most of the communication of the database server happens over Port 1433 (Microsoft default). This will be reviewed closer to deployment as well.

8.4 Disaster Recovery and Business Continuity

As detailed in section 8.3.1, the testing and production environments are a mirror of each other, which provides the basic redundancy for a disaster recovery strategy.

In addition, there are few operational considerations required to ensure the recovery is smooth and meets the needs of the business. Here are the assumptions for the recommendations to follow:

- A downtime of 2-4 hours is acceptable in case of failure.
- The respective members of the consortium have the personnel and/or Service Level Agreements (SLA) in place to guarantee that the recovery plans are executed.

Backup / Restore

All backups for the servers are made on a typical Grandfather, Father and Son rotation basis. It is recommended that there be daily backups for all the servers across all environments. Further in-line with industry trends, if possible, these backups should be available on disk storage accessible from both sites that host production and testing environments.

Database

The database logging mechanism needs to be configured so that it can support recovery for transactions made within a period of 5 seconds. This applies to the connected solutions that persist data continuously to the database. This is a standard feature available through Microsoft SQL Server and is a part of the initial setup process.

Synchronization of Environments

The testing environment serves as the gate for all releases, and hence, always has the latest code base for the system. However, it is recommended to have a copy of the OLTP database placed on a separate instance on the testing database server on a weekly basis to ensure that the testing server has the latest changes. Although Microsoft SQL Server Log Shipping is an option, it is not considered since the server backups and application of the differential backup should be sufficient given the 2-4 hour window.

Servers

Most of the servers are clustered, and hence, provide the necessary resilience against hardware failure. The Web servers however are in a farm and utilized on a round robin basis. It is recommended to have a hardware cluster for the database.

Server Monitoring

There must be some proactive server monitoring in the production environment in order to ensure that failures are averted before they happen. The market leaders are IBM Tivoli and Computer Associates Unicenter. Recommendations on the specific parts of the tool suites require a detailed evaluation of the infrastructure. However, the broad areas that the tools cover are servers, network and security.

Additional details on disaster recovery are addressed in the Disaster Recovery Plan.

9 Standards

The standards presented in this section represent currently documented standards. These may be enhanced throughout the project as standards are improved or added.

9.1 Coding Standards

9.1.1 Naming Guidelines

Of all the components that make up a coding standard, naming standards are the most visible and arguably the most important. Always use Option Explicit and keep Option Strict on.

Capitalization Styles

Use the following three conventions for capitalizing identifiers.

Pascal Case

The first letter in the identifier and the first letter of each subsequent concatenated word are capitalized. Use Pascal case for identifiers of three or more characters. For example:

Back**C**olor

Camel Case

The first letter of an identifier is lowercase and the first letter of each subsequent concatenated word is capitalized. For example:

back**C**olor

Uppercase

All letters in the identifier are capitalized. Use this convention only for identifiers that consist of two or fewer letters. For example:

System.IO
System.Web.IO

One might also have to capitalize identifiers to maintain compatibility with existing, unmanaged symbol schemes, where all uppercase characters are often used for enumerations and constant values. In general, these symbols should not be visible outside of the assembly that uses them.

The following table summarizes the capitalization rules and provides examples for the different types of identifiers.

Identifier	Case	Example
Class	Pascal	AppDomain
Enum type	Pascal	ErrorLevel
Enum values	Pascal	FatalError
Event	Pascal	ValueChange

Exception class	Pascal	WebException Note: Always ends with the suffix Exception.
Read-only Static field	Pascal	RedValue
Interface	Pascal	IDisposable Note: Interfaces always begin with the prefix I.
Method	Pascal	ToString
Namespace	Pascal	System.Drawing
Parameter	Camel	typeName
Property	Pascal	BackColor
Protected instance field	Camel	redValue Note: Rarely used. A property is preferable to using a protected instance field.
Public instance field	Pascal	RedValue Note: Rarely used. A property is preferable to using a public instance field.

Abbreviations

To avoid confusion and guarantee cross-language interoperation, follow these rules regarding the use of abbreviations:

- Do not use abbreviations or contractions as parts of identifier names. For example, use GetWindow instead of GetWin.
- Where appropriate, use well-known acronyms to replace lengthy phrase names. For example, use UI for User Interface and OLAP for On-Line Analytical Processing.
- Do not use acronyms that are not generally accepted in the computing field. (For example, XML, TTL, DNS, UI, IP and IO are all OK.)
- When using acronyms, use Pascal case or Camel case for acronyms more than two characters long. For example, use HtmlButton or HTMLButton. However, capitalize acronyms that consist of only two characters, such as System.IO instead of System.Io.
- Do not use abbreviations in identifiers or parameter names. If it is necessary to use abbreviations, use Camel case for abbreviations that consist of more than two characters, even if this contradicts the standard abbreviation of the word.

Class Naming Guidelines

The following rules outline the guidelines for naming classes:

- Use a noun or noun phrase to name a class.

- Use Pascal case.
- Use abbreviations sparingly.
- Do not use a type prefix, such as c or class, on a class name. For example, use the class name FileStream rather than CFileStream.
- Do not use the underscore character (_).
- Occasionally, it is necessary to provide a class name that begins with the letter I, even though the class is not an interface. This is appropriate as long as I is the first letter of an entire word that is a part of the class name. For example, the class name IdentityStore is appropriate.
- Where appropriate, use a compound word to name a derived class. The second part of the derived class's name should be the name of the base class. For example, ApplicationException is an appropriate name for a class derived from a class named Exception, because ApplicationException is a kind of Exception. Use reasonable judgment in applying this rule. For example, Button is an appropriate name for a class derived from Control. Although a button is a kind of control, making Control a part of the class name would lengthen the name unnecessarily.

Interface Naming Guidelines

The following rules outline the naming guidelines for interfaces:

- Name interfaces with nouns or noun phrases or adjectives that describe behavior. For example, the interface name IComponent uses a descriptive noun. The interface name ICustomAttributeProvider uses a noun phrase. The name IPersistable uses an adjective.
- Use Pascal case.
- Use abbreviations sparingly.
- Prefix interface names with the letter I to indicate that the type is an interface.
- Use similar names when defining a class/interface pair where the class is a standard implementation of the interface. The names should differ only by the letter I prefix on the interface name.
- Do not use the underscore character (_).

Attribute Naming Guidelines

Always add the suffix Attribute to custom attribute classes. The following is an example of a correctly named attribute class:

```
Public Class ObsoleteAttribute
```

Enumeration Type Naming Guidelines

The enumeration (Enum) value type inherits from the Enum *Class*. The following rules outline the naming guidelines for enumerations:

- Use Pascal case for Enum types and value names.
- Use abbreviations sparingly.

- Do not use an Enum suffix on Enum type names.
- Use a singular name for most Enum types, but use a plural name for Enum types that are bit fields.
- Always add the FlagsAttribute to a bit field Enum type.

Static Field Naming Guidelines

The following rules outline the naming guidelines for static fields:

- Use nouns, noun phrases, or abbreviations of nouns to name static fields.
- Use Pascal case.
- Use a Hungarian notation prefix on static field names.
- Use static properties instead of public static fields whenever possible.

Parameter Naming Guidelines

The following rules outline the naming guidelines for parameters:

- Use descriptive parameter names. Parameter names should be descriptive enough that the name of the parameter and its type can be used to determine its meaning in most scenarios.
- Use Camel case for parameter names.
- Use names that describe a parameter's meaning rather than names that describe a parameter's type. Development tools should provide meaningful information about a parameter's type. Therefore, a parameter's name can be put to better use by describing meaning. Use type-based parameter names sparingly and only where it is appropriate.
- Do not use reserved parameters. Reserved parameters are private parameters that might be exposed in a future version if they are needed. Instead, if more data is needed in a future version of a class library, add a new overload for a method.
- Do not prefix parameter names with Hungarian type notation.

Method Naming Guidelines

The following rules outline the naming guidelines for methods:

- Use verbs or verb phrases to name methods.
- Use Pascal case.

The following are examples of correctly named methods:

```
RemoveAll()
GetCharArray()
Invoke()
```

Property Naming Guidelines

The following rules outline the naming guidelines for properties:

- Use a noun or noun phrase to name properties.
- Use Pascal case.
- Do not use Hungarian notation.
- Consider creating a property with the same name as its underlying type. For example, if a property named Color is declared, the type of the property should likewise be Color.

Event Naming Guidelines

The following rules outline the naming guidelines for events:

- Use an EventHandler suffix on event handler names.
- Specify two parameters named sender and e. The sender parameter represents the object that raised the event. The sender parameter is always of type object, even if it is possible to use a more specific type. The state associated with the event is encapsulated in an instance of an event class named e. Use an appropriate and specific event class for the e parameter type.
- Name an event argument class with the EventArgs suffix.
- Consider naming events with a verb.
- Use a gerund (the "ing" form of a verb) to create an event name that expresses the concept of pre-event, and a past-tense verb to represent post-event. For example, a Close event that can be cancelled should have a Closing event and a Closed event. Do not use the BeforeXX/AfterXXX naming pattern.
- Do not use a prefix or suffix on the event declaration on the type. For example, use Close instead of OnClose.
- In general, provide a protected method called OnXXX on types with events that can be overridden in a derived class. This method should only have the event parameter e, because the sender is always the instance of the type.

Control Naming Guidelines

The choice to prefix design-time controls with a predetermined string is a sound one. It allows the developer to distinguish easily between design-time controls and other object kinds.

All controls must be changed from their default name to an appropriate replacement value. This assists future development and simply looks better. This must be done regardless of how insignificant the control appears.

Controls have their own set of prefixes. They are used to identify the type of control so that code can be visually checked for correctness. They also assist in making it easy to know the name of a control without continually needing to look it up.

Table of Standard Control Prefixes

The following table is a list of the common types of controls together with their prefixes:

Prefix	Control
lbl	Label
llbl	LinkLabel
btn	Button
txt	Textbox
mnu	MainMenu
chk	CheckBox
rdb	RadioButton
grp	GroupBox
pic	PictureBox
dgv	DataGridView
lst	ListBox
cbo	ComboBox
lvw	ListView
trv	TreeView
tab	TabControl
dtp	DateTimePicker
mon	MonthCalendar
sbr	ScrollBar
tmr	Timer
spl	Splitter
dud	DomainUpDown
nud	NumericUpDown
trk	TrackBar
pro	ProgressBar
rtb	RichTextBox
img	ImageList
hlp	HelpProvider
tip	ToolTip
cmn	ContextMenu
tbr	ToolBar
frm	Form

Prefix	Control
bar	StatusBar
nico	NotifyIcon
ofd	OpenFileDialog
sfd	SaveFileDialog
fd	FontDialog
cd	ColorDialog
pd	PrintDialog
ppd	PrintPreviewDialog
ppc	PrintPreviewControl
err	ErrorProvider
pdoc	PrintDocument
psd	PageSetupDialog
crv	CrystalReportViewer
pd	PrintDialog
fsw	FileSystemWatcher
log	EventLog
dire	DirectoryEntry
dirs	DirectorySearcher
msq	MessageQueue
pco	PerformanceCounter
pro	Process
ser	ServiceController
rpt	ReportDocument
ds	DataSet
olea	OleDbDataAdapter
olec	OleDbConnection
oled	OleDbCommand
sqla	SqlDbDataAdapter
sqlc	SqlDbConnection
sqld	SqlDbCommand
Bs	BindingSource
dvw	DataView

9.1.2 Coding Guidelines

1. Always use predefined types rather than the aliases in the *System* namespace.
2. Avoid putting multiple classes in a single file.
3. A single file should only contribute types to a single namespaces. Avoid having multiple namespaces in the same file.
4. Avoid files with more than 500 lines (excluding machine-generated code).
5. Avoid methods with more than 25 lines.
6. Lines should not exceed 80 characters.
7. Do not manually edit any machine generated code.
8. If modifying machine generated code, modify the format and style to match this coding standard.
9. Avoid comments that explain the obvious.
10. Code should be self-explanatory. Good code with readable variable and method names should not require comments.
11. Document only operational assumptions, algorithm insights, etc.
12. Avoid method-level documentation.
    a. Use extensive external documentation for API documentation.
    b. Use method-level comments only as tool tips for other developers.
13. Never hard-code a numeric value; always declare a constant instead.
14. Assert every assumption.
15. Make only the most necessary types public; mark others as internal.
16. Always use zero-based arrays.
17. Avoid providing explicit values for *Enums*.
18. Avoid specifying a type for an *Enum* (like *long*).
19. Never use *goto* unless in a switch statement fall-through.
20. Avoid function calls in Boolean conditional statements. Assign into local variables and check on them.
21. Always explicitly initialize an array of reference types using a Do loop.
22. Only catch exceptions for which there is explicit handling.
23. In a catch statement that throws an exception, always throw the original exception to maintain stack location of original error.
24. Avoid error code as methods return values.
25. Do not use the *New* inheritance qualifier. Use *Override* instead.
26. Minimize code in application assemblies (EXE client assemblies); use class libraries instead to contain business logic.
27. Never hardcode strings that will be presented to end users; use resources instead.
28. Never hardcode strings that might change based on deployment, such as connection strings.

29. Never use unsafe code unless using interop.
30. Always use interfaces.
31. Avoid multiple Main() methods in a single assembly.
32. Never assume a type supports an interface. Defensively query for that interface.
33. Do not provide public or protected member variables. Use properties instead.
34. Do not provide public event member variables. Use event accessors instead.
35. Classes and interfaces should have at least 2:1 ratio of methods to properties.
36. Avoid interfaces with one member.
37. Strive to have 3-5 members per interface and no more than 20 members per interface.
38. Avoid events as interface members.
39. Avoid abstract methods; use interfaces instead.
40. Expose interfaces on class hierarchies.
41. Prefer using explicit interface implementation.
42. When building a long string, use StringBuilder, not string.
43. Avoid providing methods on structures.
44. Always provide a static constructor when providing static member variables.
45. Walk through every line of code in a "white box" testing manner.
46. Avoid code that relies on an assembly running from a particular location.
47. Do not use late-binding invocation when early-binding is possible.
48. Avoid using the trinary conditional operator.
49. Do not use the *Me* reference unless invoking another constructor from within a constructor.
50. Use application logging and tracing.
51. Do not use the *MyBase* word to access base class members, except when resolving a conflict with a subclasses member of the same name or when invoking a base class constructor.
52. Implement *Dispose()* and *Finalize()* methods based on the template provided by Microsoft.
53. Avoid casting to and from *System.Object* in code that uses generics.

9.2 Database Coding Standards

9.2.1 Coding Standards

The naming conventions for database management (DBMS) objects are similar to the SQL naming standards. Specifically, object names:

- Are not case-sensitive.
- Are to be unique and not the name of another object.

- Are to be stated in the singular.
- Do not use DBMS or ANSI reserved words.
- State what the object is as a descriptive phrase or sentence.

Names should be unique, meaningful, clearly understood, and not synonyms, homonyms, or uncommon abbreviations. To avoid confusion, assign names that make it absolutely clear which kind of thing the user is dealing with. The exact meaning and interpretation of the defined object should be apparent from the name. A name should be clear enough to allow only one possible interpretation. Here is an example of a set of naming standards for tables and columns:

1. The complete entity has a simple type name reflecting the real-world object, e.g. city, company, course.
2. The corresponding external name appends name or title to the entity type name, e.g. city_name, company_name, course_title.
3. The corresponding internal identifier appends "id" or "code" to the entity type name, e.g. state_code, course_id. Alternatively where an industry standard name is in common use, it can be substituted, e.g. ISBN instead of book_id.

Table Names

Table names should describe the entity the table represents using complete English words. Names can use commonly recognized abbreviations only if the abbreviations are known agency wide. Helpful guidelines for naming a table are:

- Use an upper case letter to mark the beginning of a new word followed by lower case letters, e.g. PatientHistory instead of Patienthistory.
- Use full, descriptive, pronounceable names and avoid the use of abbreviations, e.g. Employee instead of Empl.
- Make the name singular, not plural, e.g. Book instead of Books.
- Do not use quotation marks, underscores, or spaces.
- Avoid the use of $ and #.
- Do not use a prefix in front of the name e.g. TblAgency.
- Avoid embedding specialized meaning into the name.

Column Names

Column names should describe the field they represent as precisely as possible. As with tables, column names should be complete English words. Abbreviated names should be avoided but can be used when their meanings are understood by all agency personnel. If abbreviations are used, the name should be as consistent as possible with the abbreviations used across the DBMS. Guidelines for naming columns are:

- Use lower case in the word, e.g. FreightCostAmount instead of FREIGHTCOSTAMOUNT.
- Use Pascal Case to separate words, e.g. MiddleInitial instead of middleinitial.

- Fully spell out the English word as much as possible, e.g. FirstName instead of FName.
- Use the same column names to describe the same things across tables.
- Do not use a prefix in front of the name or a suffix at the end of the name, e.g. txt_last_name.
- Be descriptive and avoid the use of synonyms, e.g. AgentName instead of Representative.
- Include the abbreviation 'dt' in the name when the data contains a date.
- Avoid the use of $ and #.

Index and Constraint Names

Index names in the database begin with a prefix indicating the type of constraint followed by the name of the table on which they are placed. For a primary key, the prefix is 'PK'. All other indexes use the 'IDX' prefix, followed by a numeric sequence. Examples of indexes are:

- PK_Employee_PK
- IDX1_Employee_IDX1
- IDX2_Employee_IDX2

Constraint names have three parts to them. The first part is the prefix 'FK', then an underscore followed by the child's table name, then an underscore followed by the parent's table name. An example of a constraint name is:

- FK_Employee_Department

View Names

A view should describe the entity the view represents, such as the primary table name, plus the reason for the view, such as finding all the locations for an agency. Complete English words are to be used or commonly recognized abbreviations, if necessary. Use an upper case letter to mark the beginning of a new word followed by lower case letters. Underscores are used to separate the words. An example of a view name is:

- Agency_Lookup

Trigger Names

A trigger name should clearly indicate the table it applies to, the before or after status, the DML commands that trigger it, and whether it is a row-level or statement-level trigger. In general, the trigger name should include as much of the table name as possible. Underscores are used to separate the words.

Since a trigger name should not exceed 30 characters in length, a standard set of abbreviations is necessary when naming triggers. Abbreviations for the DML commands are 'I' for Insert, 'U' for Update', and 'D' for Delete. Abbreviations for the before/after status are 'B' for Before and 'A' for After. To indicate a row-level trigger, use the abbreviation 'ROW'. For a statement-level trigger, use the abbreviation 'STMT'. Listed below are examples of trigger names:

- Employee_BI_Row – Table Employee Before Insert by Row
- Employee_AI_Row – Table Employee After Insert by Row
- Employee_BU_STMT – Table Employee Before Update by Statement
- Employee_AU_STMT – Table Employee After Update by Statement
- Employee_BD_Row – Table Employee Before Delete by Row
- Employee_AD_STMT – Table Employee After Delete by Statement
- Employee_BUI_Row – Table Employee Before Update Insert by Row
- Employee_BUID_STMT – Table Employee Before Update Insert Delete by Statement
- Employee_AIU_Row – Table Employee After Insert Update by Row
- Employee_AIUD_STMT – Table Employee After Insert Update Delete by Statement

Package, Function, and Procedure Names

Packages, functions, and procedures should be named according to the business function they perform or the business rule they enforced. There should be no ambiguity about their purpose. A verb must describe what it does, like the verb 'Add'. The name should also include the name of the major table(s) it impacts. If the tables are properly named, then the name of the table should be the direct object upon which the verb acts. In addition, the name should be suffixed with an indicator of what type of program object it is.

Complete English words are to be used or commonly recognized abbreviations, if necessary. Use an upper case letter to mark the beginning of a new word followed by lower case letters. Underscores are used to separate the words. Here are examples of names for functions, and procedures:

- Get_Class_Code_FUNC
- Update_Client

User Roles Names

User role names should include a word describing the amount or type of privilege, the schema name the role is used in, and the suffix 'ROLE'. Complete English words are to be used or commonly recognized abbreviations, if necessary. Use an upper case letter to mark the beginning of a new word followed by lower case letters. Underscores are used to separate the English words. Examples of role names are:

- Full_Priviledge_ORYX_ROLE
- Read_All_CMHIFL_ROLE

9.2.2 Guidelines

1. If possible, try to keep object names within a 30 byte limit.
2. Try not to use system tables directly. System table structures may change in a future release. Wherever possible, use the sp_help* stored procedures or INFORMATION_SCHEMA views. However, there will be situations in which accessing system tables is unavoidable.
3. Make sure to normalize data at least until 3rd normal form. At the same time, do not compromise on query performance. A little bit of denormalization helps queries perform faster.
4. Write comments in stored procedures, triggers, and SQL batches generously. This helps other programmers understand the code clearly. The length of the comments does not impact the performance.
5. Do not use SELECT * in queries. Always write the required column names after the SELECT statement, like SELECT CustomerID, CustomerFirstName, City. This technique results in less disk IO and less network traffic, and hence, better performance.
6. Try to avoid server-side cursors as much as possible. Always stick to 'set based approach' instead of a 'procedural approach' for accessing/manipulating data. Cursors can be easily avoided by SELECT statements in many cases. If a cursor is unavoidable, use a simple WHILE loop to loop through the table. A WHILE loop is faster than a cursor most of the time. But for a WHILE loop to replace a cursor, a column (primary key or unique key) is needed to identify each row uniquely. Every table must have a primary or unique key.
7. Avoid the creation of temporary tables while processing data, as much as possible, as creating a temporary table means more disk IO. Consider advanced SQL or views or table variables of SQL Server or derived tables instead of temporary tables. Keep in mind that, in some cases, using a temporary table performs better than a highly complicated query.
8. Try to avoid wildcard characters at the beginning of a word while searching using the LIKE keyword, as that results in an index scan, which is defeating the purpose of having an index. The following statement results in an index scan, while the second statement results in an index seek:

    - SELECT LocationID FROM Locations WHERE Specialties LIKE '%pples'
    - SELECT LocationID FROM Locations WHERE Specialties LIKE 'A%s'

    Also avoid searching with not equals operators (<> and NOT) as they result in table and index scans. If heavy text-based searches are necessary, consider using the Full-Text search feature of SQL Server for better performance.
9. While designing a database, design it keeping 'performance' in mind. Tuning performance later, when the database is in production, is not possible as it involves rebuilding tables/indexes, re-writing queries. Use the graphical execution plan in Query Analyzer or SHOWPLAN_TEXT or SHOWPLAN_ALL commands to analyze queries. Make sure queries do 'Index seeks' instead of 'Index scans' or 'Table scans'. A table scan or an index scan should be avoided where possible. (Sometimes when the table is too small or when the whole table needs to be processed, the optimizer will choose a table or index scan.)

10. Use SET NOCOUNT ON at the beginning of SQL batches, stored procedures, and triggers in production environments, as this suppresses messages like '(1 row(s) affected)' after executing INSERT, UPDATE, DELETE and SELECT statements. This in turn improves the performance of the stored procedures by reducing the network traffic.

11. Do not prefix stored procedure names with 'sp_'. The prefix sp_ is reserved for system stored procedure that ship with SQL Server. Whenever SQL Server encounters a procedure name starting with sp_, it first tries to locate the procedure in the master database. Save time in locating the stored procedure by avoiding sp_ prefix. However, there is one exception. While creating general purpose stored procedures that are called from all databases, prefix those stored procedure names with sp_ and create them in the master database.

12. Views are generally used to show specific data to specific users based on their interest. Views are also used to restrict access to the base tables by granting permission on only views. Yet another significant use of views is that they simplify queries. Incorporate frequently required complicated joins and calculations into a view to avoid repeating those joins/calculations in all queries. Instead just select from the view.

13. Use 'User Defined Data types', if a particular column repeats in a lot of tables, so that the data type of that column is consistent across all tables.

14. Do not let front-end applications query/manipulate the data directly using SELECT or INSERT/UPDATE/DELETE statements. Instead, create stored procedures, and let applications access these stored procedures. This keeps the data access clean and consistent across all the modules of the application, at the same time centralizing the business logic within the database.

15. Use char data type for a column only when the column is non-nullable. If a char column is nullable, it is treated as a fixed length column in SQL Server. So, a char(100), when NULL, will eat up 100 bytes, resulting in space wastage. So, use varchar(100) in this situation. Of course, variable length columns do have a very little processing overhead over fixed length columns. Carefully choose between char and varchar depending upon the length of the data being stored.

16. Avoid dynamic SQL statements as much as possible. Dynamic SQL tends to be slower than static SQL, as SQL Server must generate an execution plan every time at runtime. IF and CASE statements come in handy to avoid dynamic SQL. Another major disadvantage of using dynamic SQL is that it requires the users to have direct access permissions on all accessed objects like tables and views. Generally, users are given access to the stored procedures which reference the tables, but not directly on the tables. In this case, dynamic SQL will not work. Consider the following scenario, where a user named 'dSQLuser' is added to the pubs database and is granted access to a procedure named 'dSQLproc', but not on any other tables in the pubs database. The procedure dSQLproc executes a direct SELECT on titles table that works. The second statement runs the same SELECT on titles table using dynamic SQL, and it fails with the following error:

    Server: Msg 229, Level 14, State 5, Line 1

    SELECT permission denied on object 'titles', database 'pubs', owner 'dbo'.

To reproduce the above problem, use the following commands:

```
sp_addlogin 'dSQLuser'
GO
sp_defaultdb 'dSQLuser', 'pubs'
USE pubs
GO
sp_adduser 'dSQLUser', 'dSQLUser'
GO
CREATE PROC dSQLProc
AS
BEGIN
SELECT * FROM titles WHERE title_id = 'BU1032' --This works
DECLARE @str CHAR(100)
SET @str = 'SELECT * FROM titles WHERE title_id = ''BU1032'''
EXEC (@str) --This fails
END
GO
GRANT EXEC ON dSQLProc TO dSQLuser
GO
```

Now login to the pubs database using the login dSQLuser and execute the procedure dSQLproc to see the problem.

17. Consider the following drawbacks before using IDENTITY property for generating primary keys. IDENTITY is very much SQL Server specific and causes problems if it is necessary to support different database backends for the application. IDENTITY columns have other inherent problems. IDENTITY columns run out of numbers eventually. Numbers cannot be reused automatically after deleting rows. Replication and IDENTITY columns do not always get along well. Instead of IDENTITY columns, use the UNIQUEIDENTIFIER type for primary key columns.
18. Use Unicode data types like nchar, nvarchar, ntext if the database is going to store not just plain English characters, but a variety of characters used all over the world. Use these data types only when they are absolutely needed as they need twice as much space as non-unicode data types.
19. Always use a column list in INSERT statements. This helps in avoiding problems when the table structure changes (like adding a column).
20. Perform all referential integrity checks and data validations using constraints (foreign key and check constraints). These constraints are faster than triggers. Use triggers only for auditing, custom tasks, and validations that cannot be performed using these constraints. These constraints save time as well, as it is not necessary to write code for these validations and the RDBMS will do all the work.

21. Always access tables in the same order in all stored procedures/triggers consistently. This helps in avoiding deadlocks. Other things to keep in mind to avoid deadlocks are:
    - Keep transactions as short as possible.
    - Touch as little data as possible during a transaction.
    - Never wait for user input in the middle of a transaction.
    - Do not use higher level locking hints or restrictive isolation levels unless they are absolutely needed.
    - Make front-end applications deadlock-intelligent. That is, these applications should be able to resubmit the transaction in case the previous transaction fails with error 1205.
    - In applications, process all the results returned by SQL Server immediately so that the locks on the processed rows are released, hence no blocking.
22. Consider adding a @Debug parameter to stored procedures. This can be of bit data type. When a 1 is passed for this parameter, print all the intermediate results and variable contents using SELECT or PRINT statements and when 0 is passed, do not print debug information. This helps in quick debugging of stored procedures, as it is not necessary to add and remove these PRINT/SELECT statements before and after troubleshooting problems.
23. Do not call functions repeatedly within stored procedures, triggers, functions and batches. For example, it might be necessary to find the length of a string variable in many places of a procedure, but don't call the LEN function whenever it is needed. Instead, call the LEN function once, and store the result in a variable for later use.
24. Make sure stored procedures always return a value indicating the status. Standardize on the return values of stored procedures for success and failures. The RETURN statement is meant for returning the execution status only, but not data. If it is necessary to return data, use OUTPUT parameters.
25. Always check the global variable @@ERROR immediately after executing a data manipulation statement (like INSERT/UPDATE/DELETE, unless using ADO or ADO.Net to control the transaction boundaries), so that the transaction can be rolled back in case of an error (@@ERROR will be greater than 0 in case of an error). This is important, because by default, SQL Server will not rollback all the previous changes within a transaction if a particular statement fails. This behavior can be changed by executing SET XACT_ABORT ON. The @@ROWCOUNT variable also plays an important role in determining how many rows were affected by a previous data manipulation (also, retrieval) statement. Based on that, choose to commit or rollback a particular transaction.
26. Always store 4-digit years in dates (especially, when using char or int data type columns) instead of 2-digit years to avoid any confusion and problems. This is not a problem with datetime columns, as the century is stored even if a 2-digit year is specified. But it is always a good practice to specify 4-digit years even with datetime data type columns.
27. As is true with any other programming language, do not use GOTO or use it sparingly. Excessive usage of GOTO can lead to hard-to-read-and-understand code.
28. Do not forget to enforce unique constraints on alternate keys.

29. Always be consistent with the usage of case in code. On a case insensitive server, code might work fine, but it will fail on a case sensitive SQL Server if the code is not consistent in case. For example, if there is a table in SQL Server or a database that has a case-sensitive or binary sort order, all references to the table must use the same case specified in the CREATE TABLE statement. If the table is named 'MyTable' in the CREATE TABLE statement and 'mytable' is used in the SELECT statement, an 'object not found' or 'invalid object name' error occurs.

30. Though T-SQL has no concept of constants (like the ones in VB6 or VB.Net), variables will serve the same purpose. Using variables instead of constant values within SQL statements improves readability and maintainability of the code.

31. Do not use the column numbers in the ORDER BY clause as it impairs the readability of the SQL statement. Further, changing the order of columns in the SELECT list has no impact on the ORDER BY when the columns are referred to by names instead of numbers. Consider the following example, in which the second query is more readable than the first one:

    ```
    SELECT OrderID, OrderDate
    FROM Orders
    ORDER BY 2

    SELECT OrderID, OrderDate
    FROM Orders
    ORDER BY OrderDate
    ```

32. If it is necessary to store integer data from 0 through 255, use the TINYINT data type. This data type uses only one byte to store its value, in comparison with two bytes, four bytes and eight bytes used to store the columns with the SMALLINT, INT and BIGINT data types, respectively.

33. If it is necessary to store integer data from –32,768 through 32,767, use the SMALLINT data type

34. If it is necessary to store integer data from –2,147,483,648 through 2,147,483,647, use the INT data type.

35. Use the SMALLMONEY data type instead of MONEY data type if it is necessary to store monetary data values from -214,748.3648 through 214,748.3647.

36. Use the SMALLDATE data type instead of DATETIME data type if it is necessary to store the date and time data from January 1, 1900 through June 6, 2079 with accuracy to the minute.

9.2.3 Security

1. Never give users direct permissions to tables.

2. Assign specific roles to execute stored procedures in the database and then add users to that role.

3. Always verify the usage of external stored procedures in the system, making sure they do not require special logins or security privileges, which might make the database server vulnerable

10 Appendix A: Data Dictionary and Data Models

Accompanying this document are a number of supplemental files which contain detailed information about the data structures of the database.

The Data Dictionary is an Excel file (08 DTSD Data Dictionary.xls) with two tabs. The Tables tab lists all the tables in the database with a description and a mapping to the primary functional area of the system which uses that table. The Columns tab lists all the individual columns in the database by table and the details of each column (data type, nulls allowed, key indicator, and description).

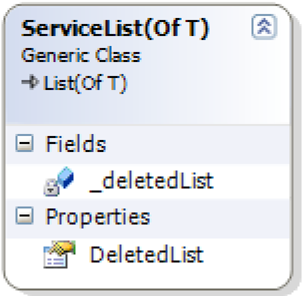
The Data Models are contained in sixteen Visio files which graphically describe all of the areas of the MPSC database (delivered in a zip file, 08 DTSD Data Models.zip). For each table depicted, the model includes the column names within the table. The data models are split across multiple files in order to support printing in a readable fashion on paper no larger than 11" x 17". Each data model includes a legend that references the scope of the model, the file name, and when it was last updated. The data models show the cardinality of the relationships and the foreign keys to other tables in the database. If the other tables are a part of that same data model, connection lines are also shown to depict the foreign key relationships. The files comprising the MPSC data model are:

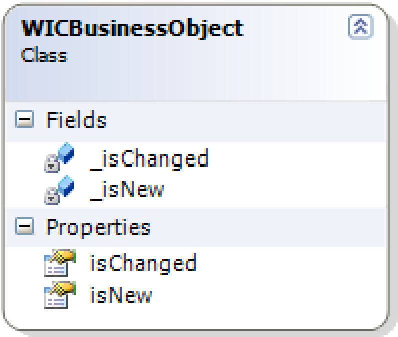
- CS 0 1 2 Intake Activity.vsd
- CS 3 4 5 Assessment.vsd
- CS 3c Interview and risk.vsd
- CS 6 Food Benefits.vsd
- FM 1 Food management.vsd
- FN 1 2.vsd
- OP 1 Staff.vsd
- OP 2 Inventory.vsd
- SA 1 System Wide.vsd
- SA 2 Clinic Services.vsd
- SA 3 Operations.vsd
- SC 0 1 Scheduler.vsd
- VM 0 1 2 Details Application.vsd
- VM 3 Oversight.vsd
- VM 4 5 6 Training Food Associations.vsd
- VM 7 8 Price Survey High Risk.vsd

11 Appendix B: Class Diagrams

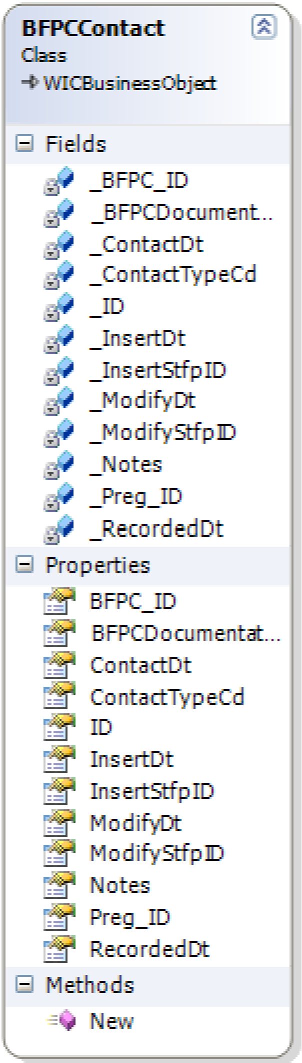
The following class diagrams cover the MPSC business classes. Each diagram shows a separate class with its respective fields and properties.

11.1 Base





11.2 Clinic Services

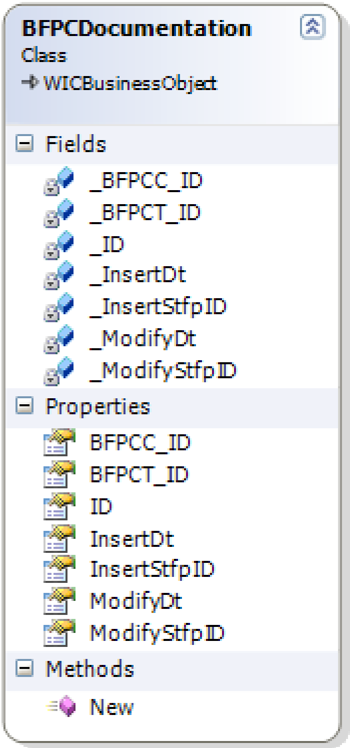


BFPCContact
Class
→ WICBusinessObject

Fields
- _BFPC_ID
- _BFPCDocument...
- _ContactDt
- _ContactTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Notes
- _Preg_ID
- _RecordedDt

Properties
- BFPC_ID
- BFPCDocumentat...
- ContactDt
- ContactTypeCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Notes
- Preg_ID
- RecordedDt

Methods
- New

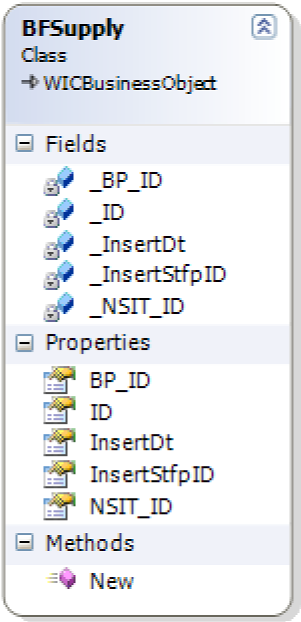


BFPCDocumentation
Class
→ WICBusinessObject

Fields
- _BFPCC_ID
- _BFPCT_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID

Properties
- BFPCC_ID
- BFPCT_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID

Methods
- New

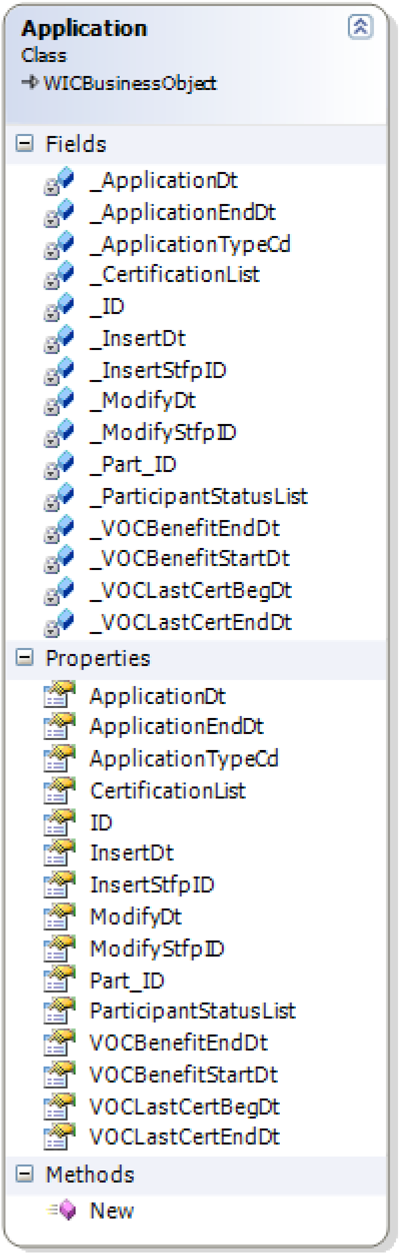


BFSupply
Class
→ WICBusinessObject

Fields
- _BP_ID
- _ID
- _InsertDt
- _InsertStfpID
- _NSIT_ID

Properties
- BP_ID
- ID
- InsertDt
- InsertStfpID
- NSIT_ID

Methods
- New



Application
Class
→ WICBusinessObject

Fields

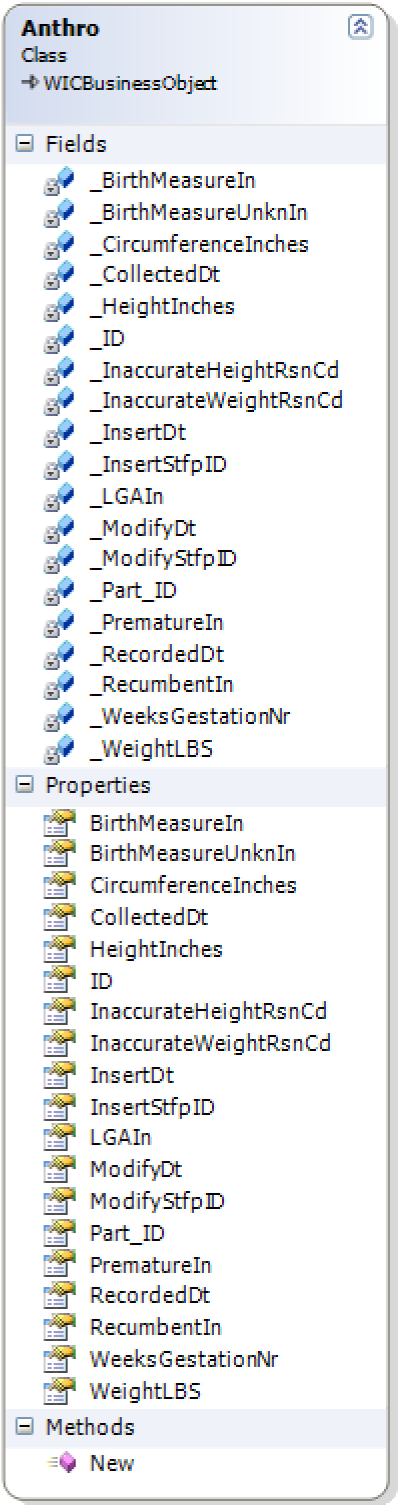
_ApplicationDt
_ApplicationEndDt
_ApplicationTypeCd
_CertificationList
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_Part_ID
_ParticipantStatusList
_VOCBenefitEndDt
_VOCBenefitStartDt
_VOCLastCertBegDt
_VOCLastCertEndDt

Properties

ApplicationDt
ApplicationEndDt
ApplicationTypeCd
CertificationList
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Part_ID
ParticipantStatusList
VOCBenefitEndDt
VOCBenefitStartDt
VOCLastCertBegDt
VOCLastCertEndDt

Methods

New



Anthro
Class
→ WICBusinessObject

Fields

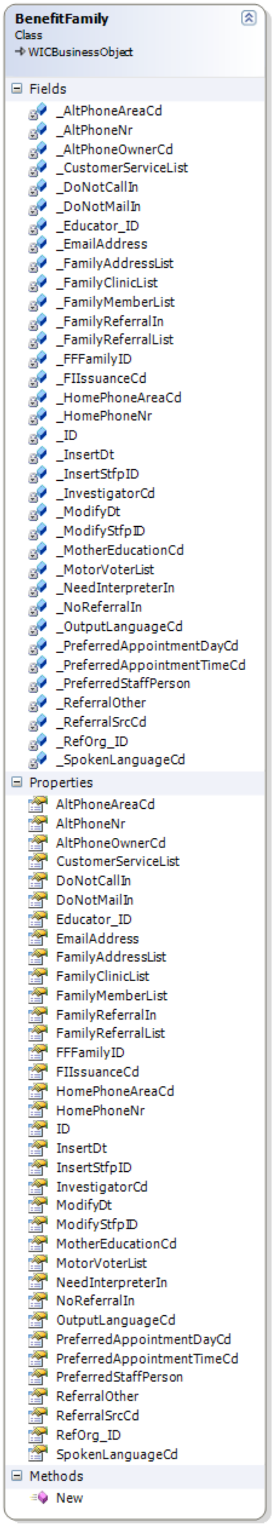
_BirthMeasureIn
_BirthMeasureUnknIn
_CircumferenceInches
_CollectedDt
_HeightInches
_ID
_InaccurateHeightRsnCd
_InaccurateWeightRsnCd
_InsertDt
_InsertStfpID
_LGAIn
_ModifyDt
_ModifyStfpID
_Part_ID
_PrematureIn
_RecordedDt
_RecumbentIn
_WeeksGestationNr
_WeightLBS

Properties

BirthMeasureIn
BirthMeasureUnknIn
CircumferenceInches
CollectedDt
HeightInches
ID
InaccurateHeightRsnCd
InaccurateWeightRsnCd
InsertDt
InsertStfpID
LGAIn
ModifyDt
ModifyStfpID
Part_ID
PrematureIn
RecordedDt
RecumbentIn
WeeksGestationNr
WeightLBS

Methods

New

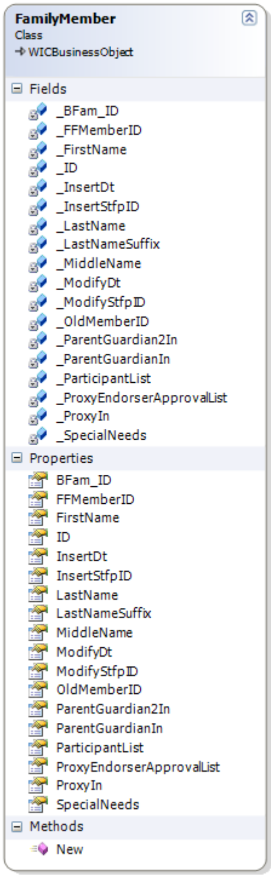


BenefitFamily
Class
→ WICBusinessObject

Fields
_AltPhoneAreaCd
_AltPhoneNr
_AltPhoneOwnerCd
_CustomerServiceList
_DoNotCallIn
_DoNotMailIn
_Educator_ID
_EmailAddress
_FamilyAddressList
_FamilyClinicList
_FamilyMemberList
_FamilyReferralIn
_FamilyReferralList
_FFFamilyID
_FIIssuanceCd
_HomePhoneAreaCd
_HomePhoneNr
_ID
_InsertDt
_InsertStfpID
_InvestigatorCd
_ModifyDt
_ModifyStfpID
_MotherEducationCd
_MotorVoterList
_NeedInterpreterIn
_NoReferralIn
_OutputLanguageCd
_PreferredAppointmentDayCd
_PreferredAppointmentTimeCd
_PreferredStaffPerson
_ReferralOther
_ReferralSrcCd
_RefOrg_ID
_SpokenLanguageCd

Properties
AltPhoneAreaCd
AltPhoneNr
AltPhoneOwnerCd
CustomerServiceList
DoNotCallIn
DoNotMailIn
Educator_ID
EmailAddress
FamilyAddressList
FamilyClinicList
FamilyMemberList
FamilyReferralIn
FamilyReferralList
FFFamilyID
FIIssuanceCd
HomePhoneAreaCd
HomePhoneNr
ID
InsertDt
InsertStfpID
InvestigatorCd
ModifyDt
ModifyStfpID
MotherEducationCd
MotorVoterList
NeedInterpreterIn
NoReferralIn
OutputLanguageCd
PreferredAppointmentDayCd
PreferredAppointmentTimeCd
PreferredStaffPerson
ReferralOther
ReferralSrcCd
RefOrg_ID
SpokenLanguageCd

Methods
New

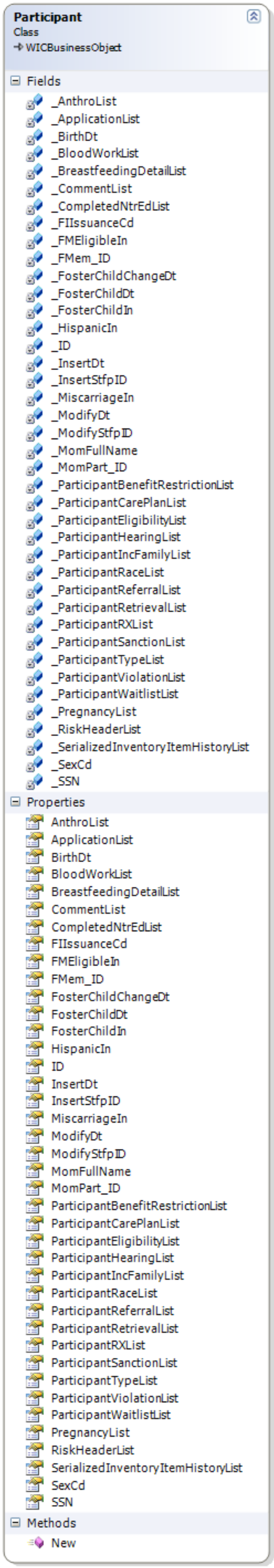


FamilyMember
Class
→ WICBusinessObject

Fields
_BFam_ID
_FFMemberID
_FirstName
_ID
_InsertDt
_InsertStfpID
_LastName
_LastNameSuffix
_MiddleName
_ModifyDt
_ModifyStfpID
_OldMemberID
_ParentGuardian2In
_ParentGuardianIn
_ParticipantList
_ProxyEndorserApprovalList
_ProxyIn
_SpecialNeeds

Properties
BFam_ID
FFMemberID
FirstName
ID
InsertDt
InsertStfpID
LastName
LastNameSuffix
MiddleName
ModifyDt
ModifyStfpID
OldMemberID
ParentGuardian2In
ParentGuardianIn
ParticipantList
ProxyEndorserApprovalList
ProxyIn
SpecialNeeds

Methods
New

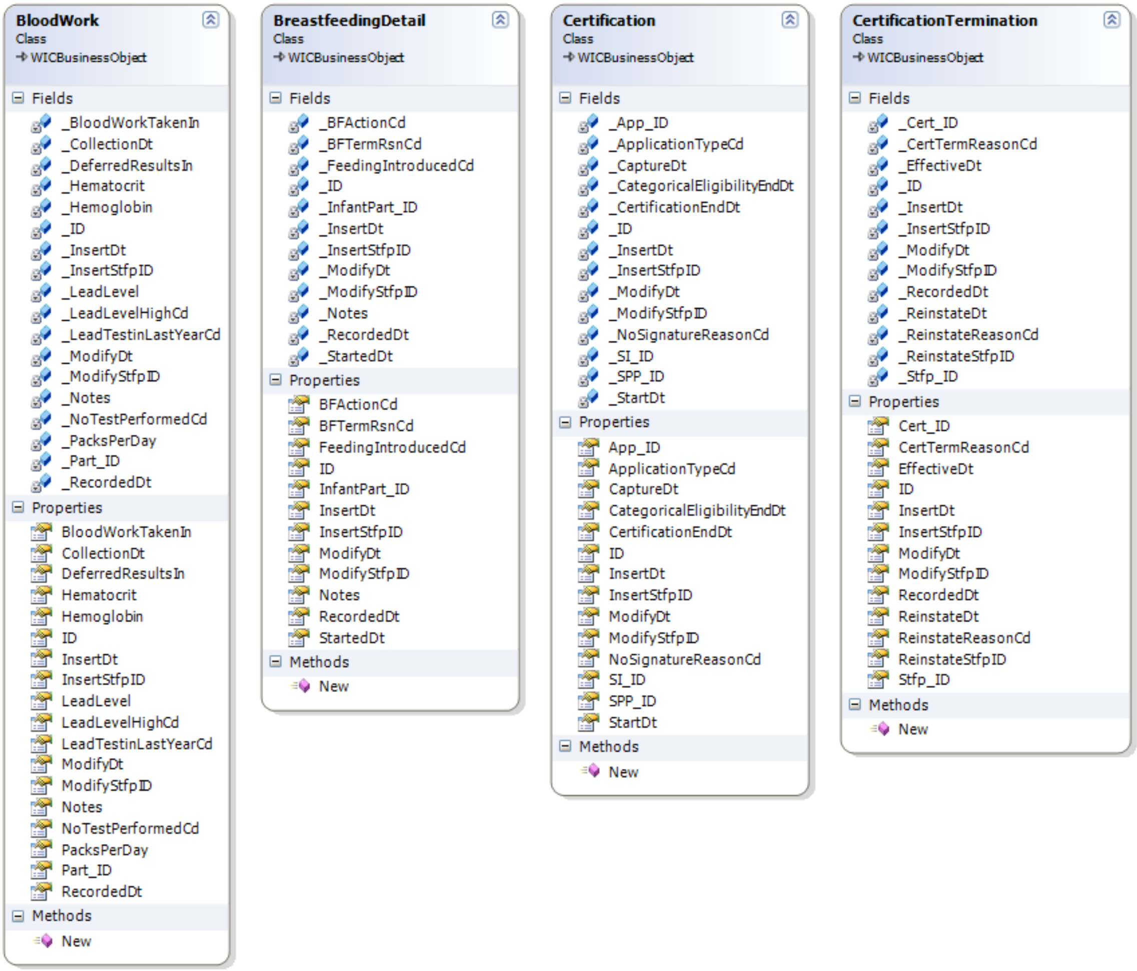


Participant
Class
→ WICBusinessObject

Fields
_AnthroList
_ApplicationList
_BirthDt
_BloodWorkList
_BreastfeedingDetailList
_CommentList
_CompletedNtrEdList
_FIIssuanceCd
_FMEligibleIn
_FMem_ID
_FosterChildChangeDt
_FosterChildDt
_FosterChildIn
_HispanicIn
_ID
_InsertDt
_InsertStfpID
_MiscarriageIn
_ModifyDt
_ModifyStfpID
_MomFullName
_MomPart_ID
_ParticipantBenefitRestrictionList
_ParticipantCarePlanList
_ParticipantEligibilityList
_ParticipantHearingList
_ParticipantIncFamilyList
_ParticipantRaceList
_ParticipantReferralList
_ParticipantRetrievalList
_ParticipantRXList
_ParticipantSanctionList
_ParticipantTypeList
_ParticipantViolationList
_ParticipantWaitlistList
_PregnancyList
_RiskHeaderList
_SerializedInventoryItemHistoryList
_SexCd
_SSN

Properties
AnthroList
ApplicationList
BirthDt
BloodWorkList
BreastfeedingDetailList
CommentList
CompletedNtrEdList
FIIssuanceCd
FMEligibleIn
FMem_ID
FosterChildChangeDt
FosterChildDt
FosterChildIn
HispanicIn
ID
InsertDt
InsertStfpID
MiscarriageIn
ModifyDt
ModifyStfpID
MomFullName
MomPart_ID
ParticipantBenefitRestrictionList
ParticipantCarePlanList
ParticipantEligibilityList
ParticipantHearingList
ParticipantIncFamilyList
ParticipantRaceList
ParticipantReferralList
ParticipantRetrievalList
ParticipantRXList
ParticipantSanctionList
ParticipantTypeList
ParticipantViolationList
ParticipantWaitlistList
PregnancyList
RiskHeaderList
SerializedInventoryItemHistoryList
SexCd
SSN

Methods
New



BloodWork
Class
→ WICBusinessObject

Fields
- _BloodWorkTakenIn
- _CollectionDt
- _DeferredResultsIn
- _Hematocrit
- _Hemoglobin
- _ID
- _InsertDt
- _InsertStfpID
- _LeadLevel
- _LeadLevelHighCd
- _LeadTestinLastYearCd
- _ModifyDt
- _ModifyStfpID
- _Notes
- _NoTestPerformedCd
- _PacksPerDay
- _Part_ID
- _RecordedDt

Properties
- BloodWorkTakenIn
- CollectionDt
- DeferredResultsIn
- Hematocrit
- Hemoglobin
- ID
- InsertDt
- InsertStfpID
- LeadLevel
- LeadLevelHighCd
- LeadTestinLastYearCd
- ModifyDt
- ModifyStfpID
- Notes
- NoTestPerformedCd
- PacksPerDay
- Part_ID
- RecordedDt

Methods
- New

BreastfeedingDetail
Class
→ WICBusinessObject

Fields
- _BFActionCd
- _BFTermRsnCd
- _FeedingIntroducedCd
- _ID
- _InfantPart_ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Notes
- _RecordedDt
- _StartedDt

Properties
- BFActionCd
- BFTermRsnCd
- FeedingIntroducedCd
- ID
- InfantPart_ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Notes
- RecordedDt
- StartedDt

Methods
- New

Certification
Class
→ WICBusinessObject

Fields
- _App_ID
- _ApplicationTypeCd
- _CaptureDt
- _CategoricalEligibilityEndDt
- _CertificationEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NoSignatureReasonCd
- _SI_ID
- _SPP_ID
- _StartDt

Properties
- App_ID
- ApplicationTypeCd
- CaptureDt
- CategoricalEligibilityEndDt
- CertificationEndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NoSignatureReasonCd
- SI_ID
- SPP_ID
- StartDt

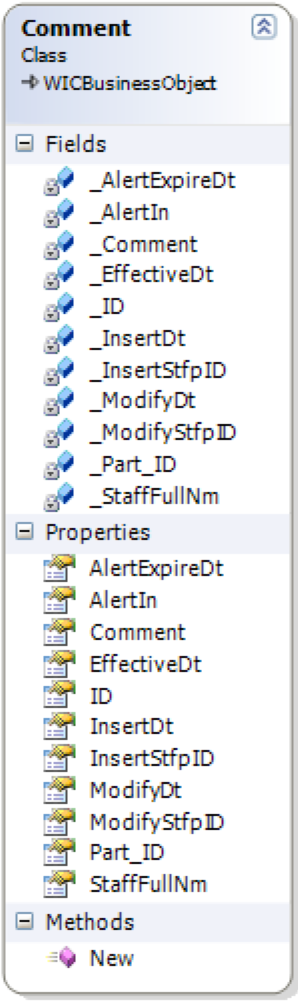
Methods
- New

CertificationTermination
Class
→ WICBusinessObject

Fields
- _Cert_ID
- _CertTermReasonCd
- _EffectiveDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _ReinstateDt
- _ReinstateReasonCd
- _ReinstateStfpID
- _Stfp_ID

Properties
- Cert_ID
- CertTermReasonCd
- EffectiveDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- ReinstateDt
- ReinstateReasonCd
- ReinstateStfpID
- Stfp_ID

Methods
- New

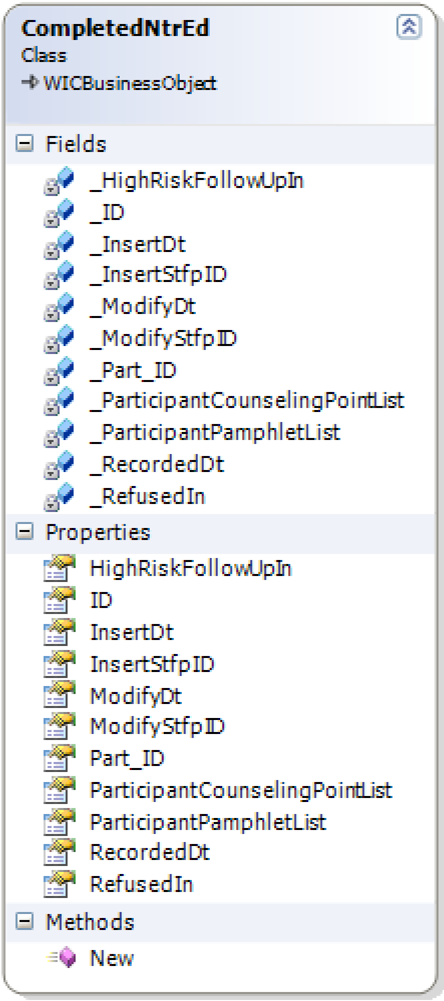


Comment
Class
→ WICBusinessObject

Fields
- _AlertExpireDt
- _AlertIn
- _Comment
- _EffectiveDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Part_ID
- _StaffFullNm

Properties
- AlertExpireDt
- AlertIn
- Comment
- EffectiveDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Part_ID
- StaffFullNm

Methods
- New

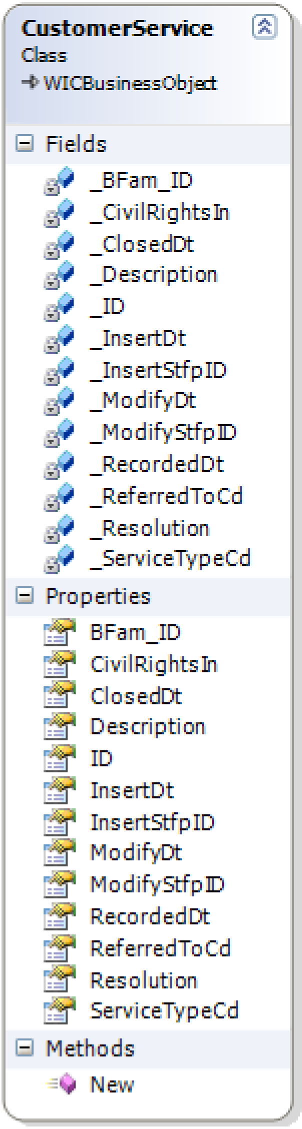


CompletedNtrEd
Class
→ WICBusinessObject

Fields
- _HighRiskFollowUpIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Part_ID
- _ParticipantCounselingPointList
- _ParticipantPamphletList
- _RecordedDt
- _RefusedIn

Properties
- HighRiskFollowUpIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Part_ID
- ParticipantCounselingPointList
- ParticipantPamphletList
- RecordedDt
- RefusedIn

Methods
- New

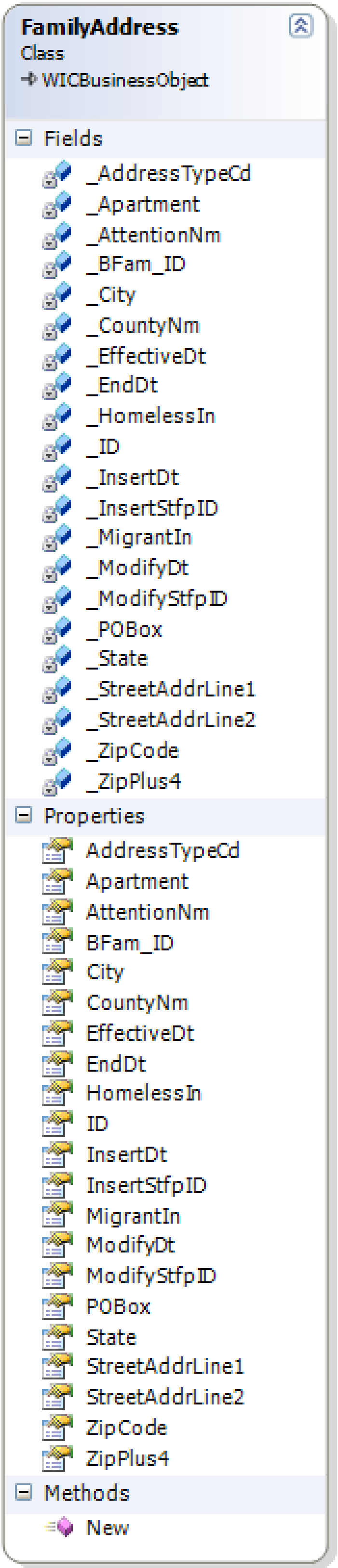


CustomerService
Class
→ WICBusinessObject

Fields
- _BFam_ID
- _CivilRightsIn
- _ClosedDt
- _Description
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _ReferredToCd
- _Resolution
- _ServiceTypeCd

Properties
- BFam_ID
- CivilRightsIn
- ClosedDt
- Description
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- ReferredToCd
- Resolution
- ServiceTypeCd

Methods
- New

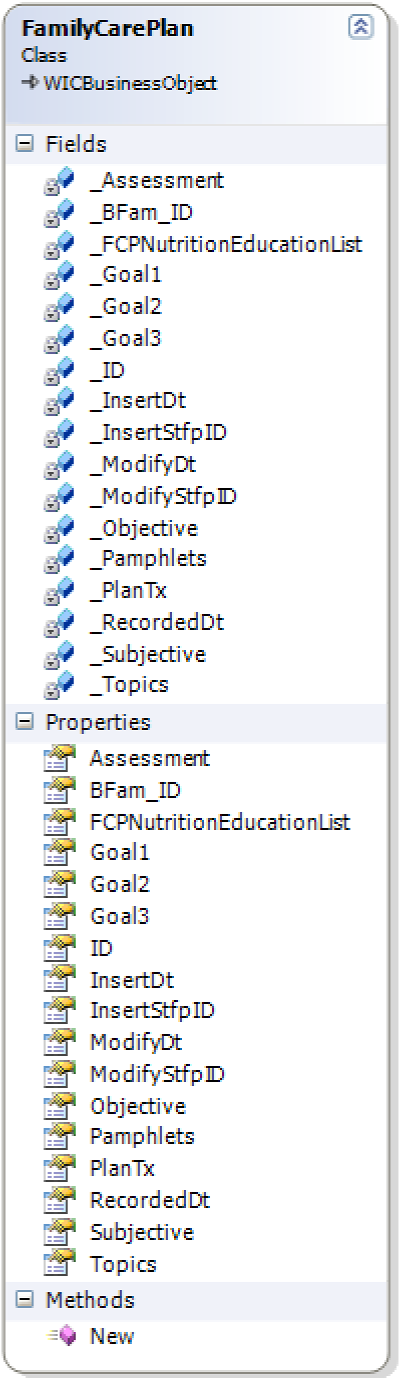


FamilyAddress
Class
WICBusinessObject

Fields
- _AddressTypeCd
- _Apartment
- _AttentionNm
- _BFam_ID
- _City
- _CountyNm
- _EffectiveDt
- _EndDt
- _HomelessIn
- _ID
- _InsertDt
- _InsertStfpID
- _MigrantIn
- _ModifyDt
- _ModifyStfpID
- _POBox
- _State
- _StreetAddrLine1
- _StreetAddrLine2
- _ZipCode
- _ZipPlus4

Properties
- AddressTypeCd
- Apartment
- AttentionNm
- BFam_ID
- City
- CountyNm
- EffectiveDt
- EndDt
- HomelessIn
- ID
- InsertDt
- InsertStfpID
- MigrantIn
- ModifyDt
- ModifyStfpID
- POBox
- State
- StreetAddrLine1
- StreetAddrLine2
- ZipCode
- ZipPlus4

Methods
- New

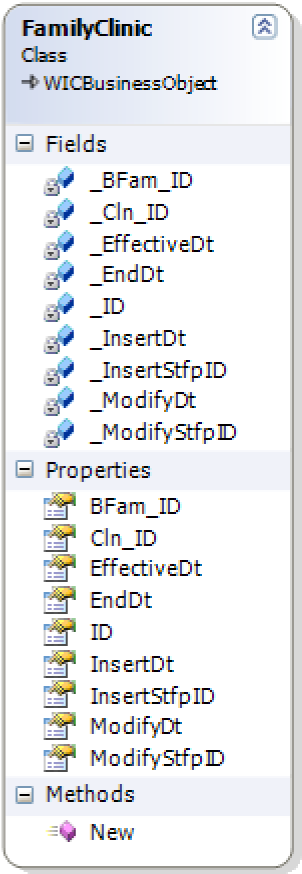


FamilyCarePlan
Class
WICBusinessObject

Fields
- _Assessment
- _BFam_ID
- _FCPNutritionEducationList
- _Goal1
- _Goal2
- _Goal3
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Objective
- _Pamphlets
- _PlanTx
- _RecordedDt
- _Subjective
- _Topics

Properties
- Assessment
- BFam_ID
- FCPNutritionEducationList
- Goal1
- Goal2
- Goal3
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Objective
- Pamphlets
- PlanTx
- RecordedDt
- Subjective
- Topics

Methods
- New

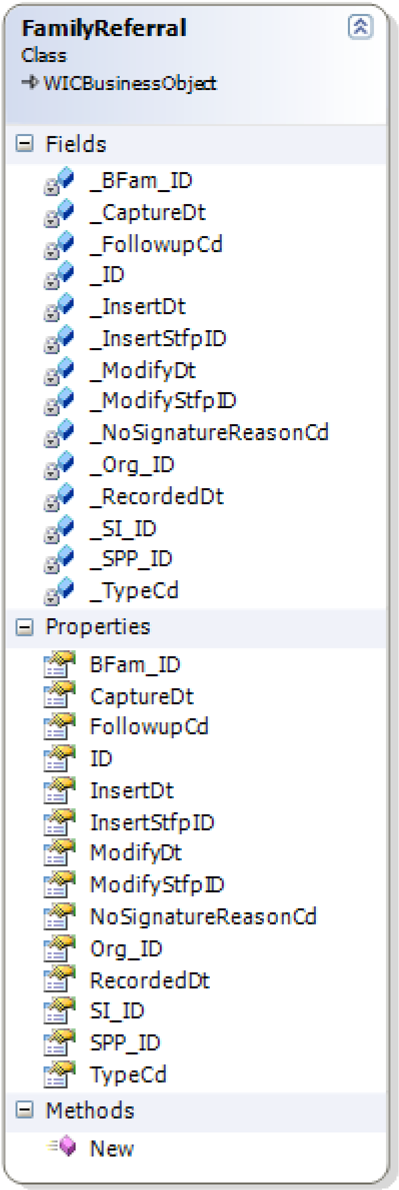


FamilyClinic
Class
WICBusinessObject

Fields
- _BFam_ID
- _Cln_ID
- _EffectiveDt
- _EndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID

Properties
- BFam_ID
- Cln_ID
- EffectiveDt
- EndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID

Methods
- New

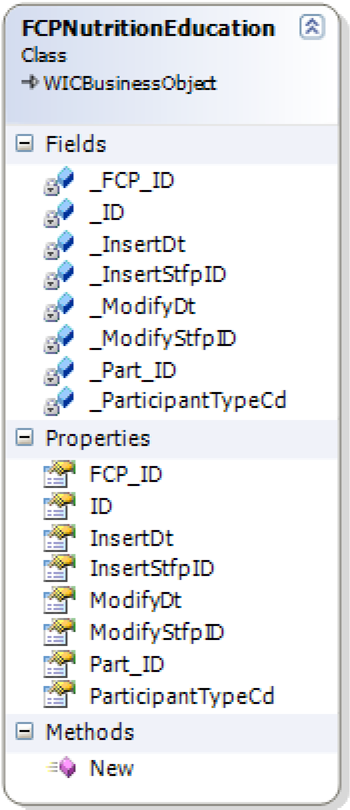


FamilyReferral
Class
→ WICBusinessObject

Fields
- _BFam_ID
- _CaptureDt
- _FollowupCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NoSignatureReasonCd
- _Org_ID
- _RecordedDt
- _SI_ID
- _SPP_ID
- _TypeCd

Properties
- BFam_ID
- CaptureDt
- FollowupCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NoSignatureReasonCd
- Org_ID
- RecordedDt
- SI_ID
- SPP_ID
- TypeCd

Methods
- New

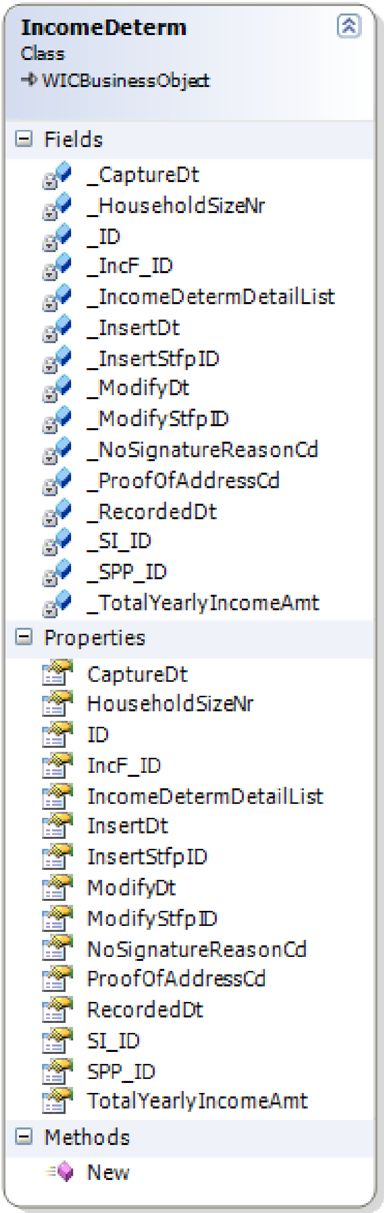


FCPNutritionEducation
Class
→ WICBusinessObject

Fields
- _FCP_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Part_ID
- _ParticipantTypeCd

Properties
- FCP_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Part_ID
- ParticipantTypeCd

Methods
- New

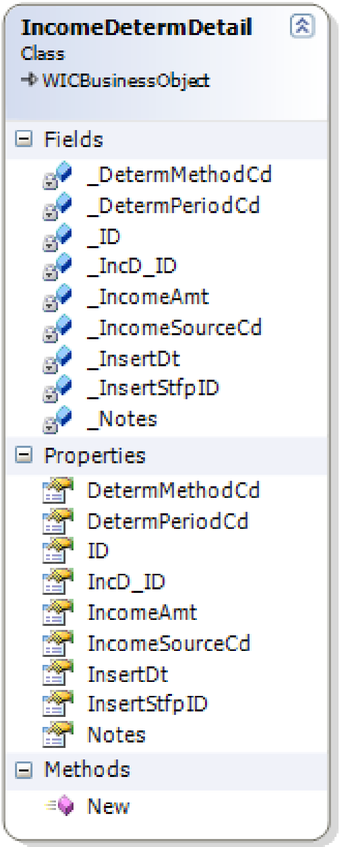


IncomeDeterm
Class
→ WICBusinessObject

Fields
- _CaptureDt
- _HouseholdSizeNr
- _ID
- _IncF_ID
- _IncomeDetermDetailList
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NoSignatureReasonCd
- _ProofOfAddressCd
- _RecordedDt
- _SI_ID
- _SPP_ID
- _TotalYearlyIncomeAmt

Properties
- CaptureDt
- HouseholdSizeNr
- ID
- IncF_ID
- IncomeDetermDetailList
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NoSignatureReasonCd
- ProofOfAddressCd
- RecordedDt
- SI_ID
- SPP_ID
- TotalYearlyIncomeAmt

Methods
- New

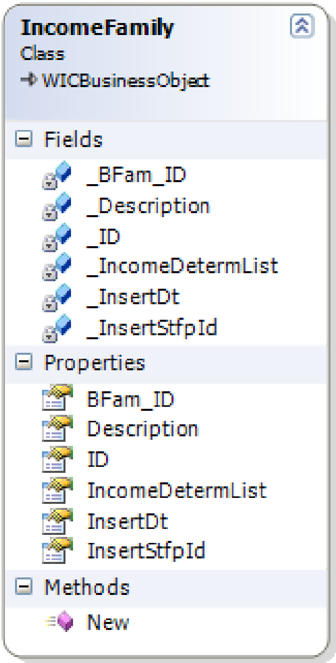


IncomeDetermDetail
Class
→ WICBusinessObject

Fields
- _DetermMethodCd
- _DetermPeriodCd
- _ID
- _IncD_ID
- _IncomeAmt
- _IncomeSourceCd
- _InsertDt
- _InsertStfpID
- _Notes

Properties
- DetermMethodCd
- DetermPeriodCd
- ID
- IncD_ID
- IncomeAmt
- IncomeSourceCd
- InsertDt
- InsertStfpID
- Notes

Methods
- New

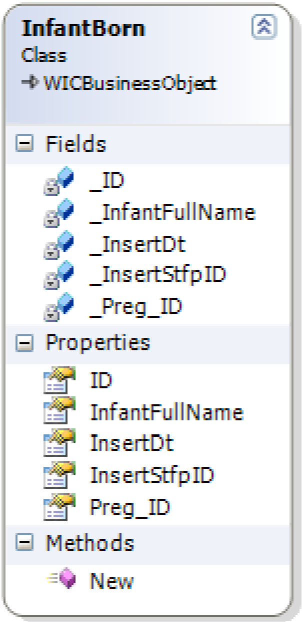


IncomeFamily
Class
→ WICBusinessObject

Fields
- _BFam_ID
- _Description
- _ID
- _IncomeDetermList
- _InsertDt
- _InsertStfpId

Properties
- BFam_ID
- Description
- ID
- IncomeDetermList
- InsertDt
- InsertStfpId

Methods
- New

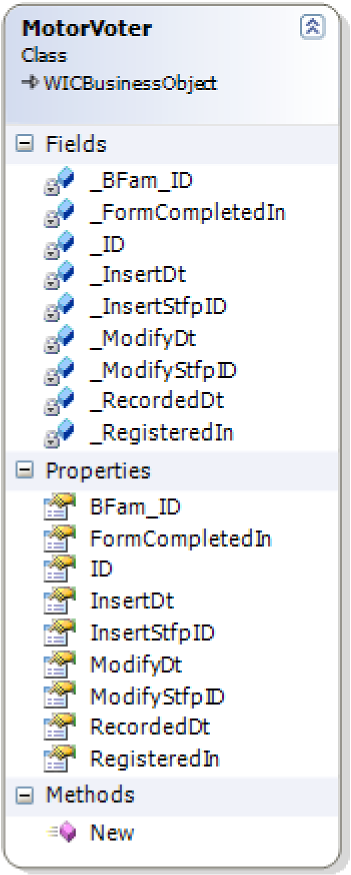


InfantBorn
Class
→ WICBusinessObject

Fields
- _ID
- _InfantFullName
- _InsertDt
- _InsertStfpID
- _Preg_ID

Properties
- ID
- InfantFullName
- InsertDt
- InsertStfpID
- Preg_ID

Methods
- New

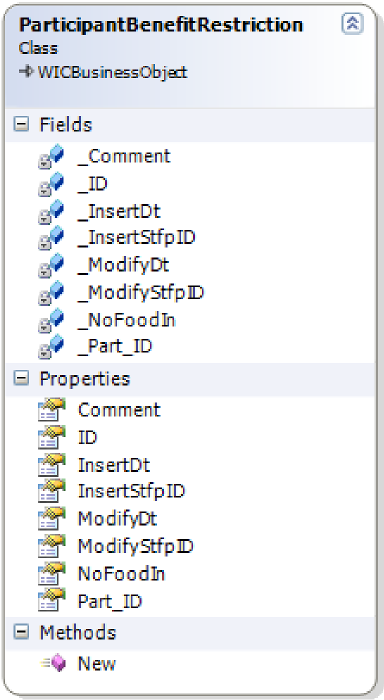


MotorVoter
Class
→ WICBusinessObject

Fields
- _BFam_ID
- _FormCompletedIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _RegisteredIn

Properties
- BFam_ID
- FormCompletedIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- RegisteredIn

Methods
- New

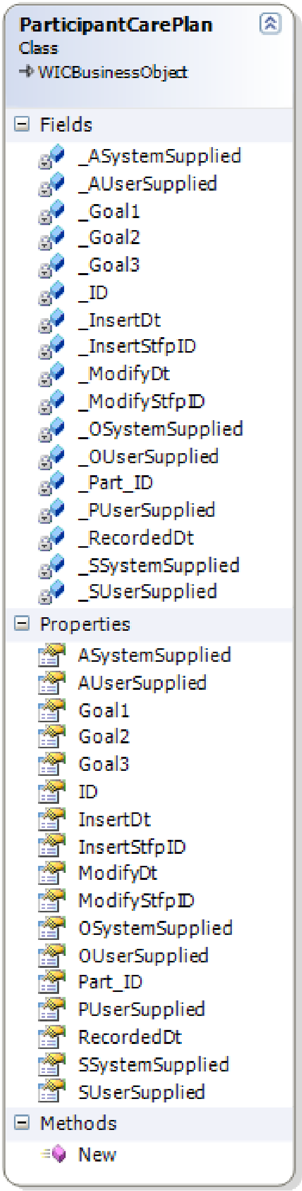


ParticipantBenefitRestriction
Class
→ WICBusinessObject

Fields
- _Comment
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NoFoodIn
- _Part_ID

Properties
- Comment
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NoFoodIn
- Part_ID

Methods
- New

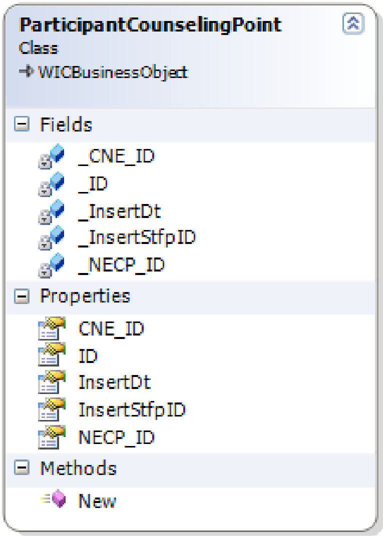


ParticipantCarePlan
Class
→ WICBusinessObject

Fields
- _ASystemSupplied
- _AUserSupplied
- _Goal1
- _Goal2
- _Goal3
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OSystemSupplied
- _OUserSupplied
- _Part_ID
- _PUserSupplied
- _RecordedDt
- _SSystemSupplied
- _SUserSupplied

Properties
- ASystemSupplied
- AUserSupplied
- Goal1
- Goal2
- Goal3
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OSystemSupplied
- OUserSupplied
- Part_ID
- PUserSupplied
- RecordedDt
- SSystemSupplied
- SUserSupplied

Methods
- New

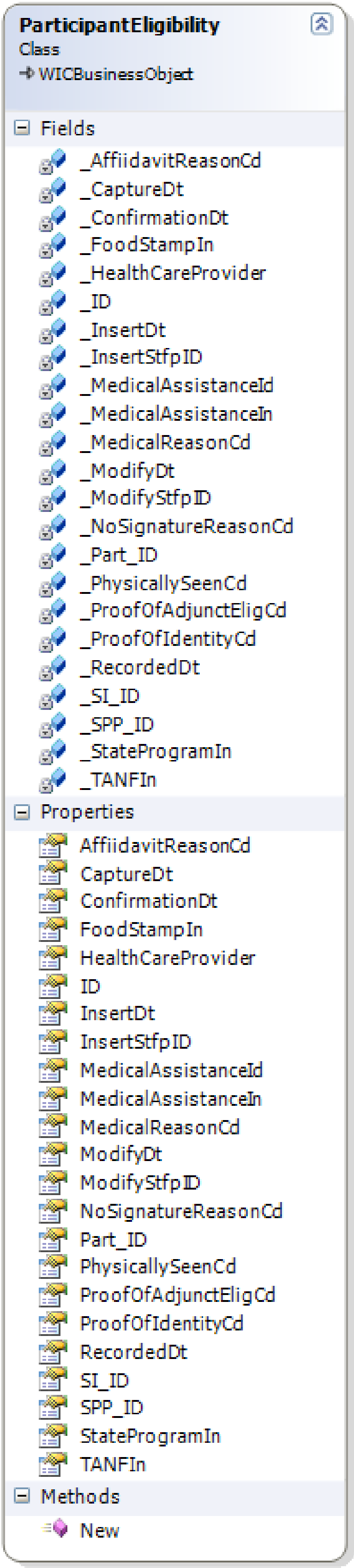


ParticipantCounselingPoint
Class
→ WICBusinessObject

Fields
- _CNE_ID
- _ID
- _InsertDt
- _InsertStfpID
- _NECP_ID

Properties
- CNE_ID
- ID
- InsertDt
- InsertStfpID
- NECP_ID

Methods
- New



ParticipantEligibility
Class
→ WICBusinessObject

⊟ Fields

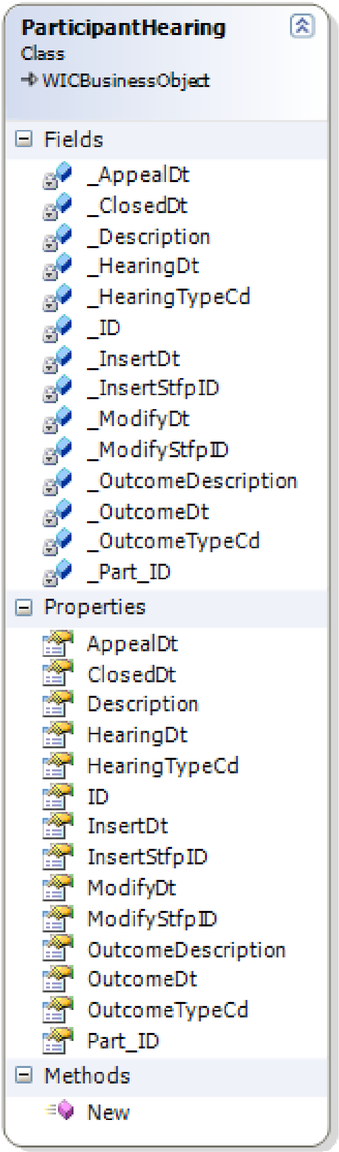
- _AffiidavitReasonCd
- _CaptureDt
- _ConfirmationDt
- _FoodStampIn
- _HealthCareProvider
- _ID
- _InsertDt
- _InsertStfpID
- _MedicalAssistanceId
- _MedicalAssistanceIn
- _MedicalReasonCd
- _ModifyDt
- _ModifyStfpID
- _NoSignatureReasonCd
- _Part_ID
- _PhysicallySeenCd
- _ProofOfAdjunctEligCd
- _ProofOfIdentityCd
- _RecordedDt
- _SI_ID
- _SPP_ID
- _StateProgramIn
- _TANFIn

⊟ Properties

- AffiidavitReasonCd
- CaptureDt
- ConfirmationDt
- FoodStampIn
- HealthCareProvider
- ID
- InsertDt
- InsertStfpID
- MedicalAssistanceId
- MedicalAssistanceIn
- MedicalReasonCd
- ModifyDt
- ModifyStfpID
- NoSignatureReasonCd
- Part_ID
- PhysicallySeenCd
- ProofOfAdjunctEligCd
- ProofOfIdentityCd
- RecordedDt
- SI_ID
- SPP_ID
- StateProgramIn
- TANFIn

⊟ Methods

- New



ParticipantHearing
Class
→ WICBusinessObject

⊟ Fields

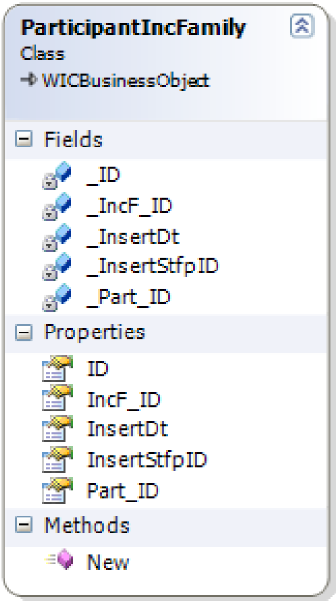
- _AppealDt
- _ClosedDt
- _Description
- _HearingDt
- _HearingTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OutcomeDescription
- _OutcomeDt
- _OutcomeTypeCd
- _Part_ID

⊟ Properties

- AppealDt
- ClosedDt
- Description
- HearingDt
- HearingTypeCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OutcomeDescription
- OutcomeDt
- OutcomeTypeCd
- Part_ID

⊟ Methods

- New



ParticipantIncFamily
Class
→ WICBusinessObject

⊟ Fields

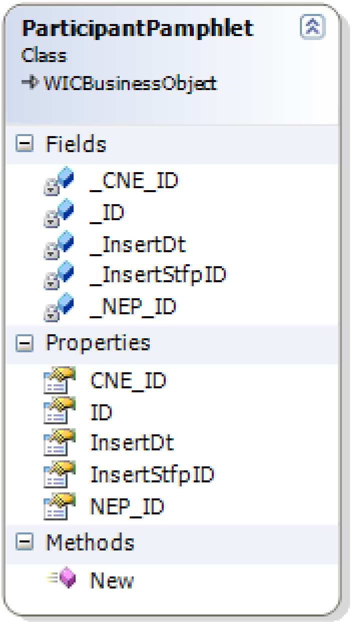
- _ID
- _IncF_ID
- _InsertDt
- _InsertStfpID
- _Part_ID

⊟ Properties

- ID
- IncF_ID
- InsertDt
- InsertStfpID
- Part_ID

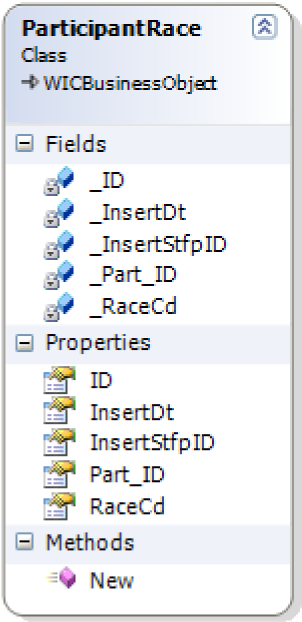
⊟ Methods

- New



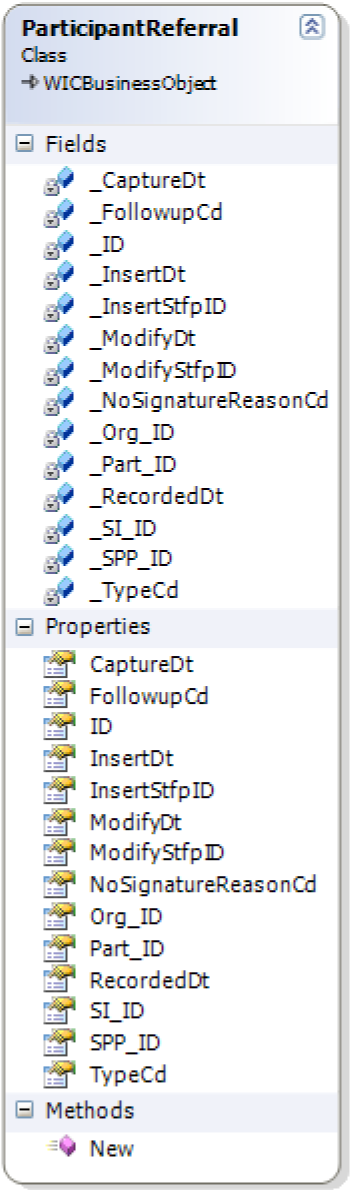
ParticipantPamphlet
Class
→ WICBusinessObject

- Fields
  - _CNE_ID
  - _ID
  - _InsertDt
  - _InsertStfpID
  - _NEP_ID
- Properties
  - CNE_ID
  - ID
  - InsertDt
  - InsertStfpID
  - NEP_ID
- Methods
  - New



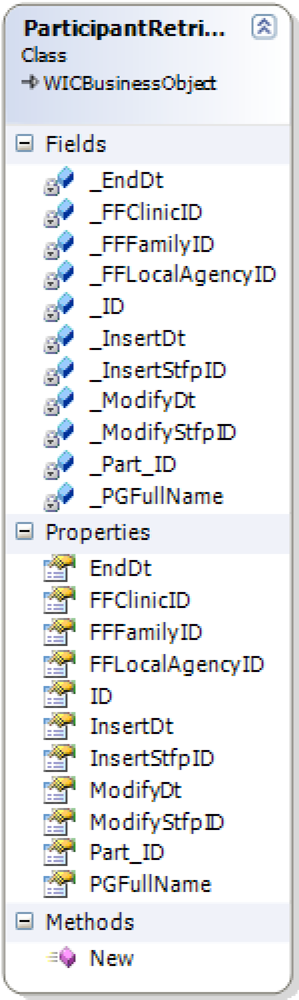
ParticipantRace
Class
→ WICBusinessObject

- Fields
  - _ID
  - _InsertDt
  - _InsertStfpID
  - _Part_ID
  - _RaceCd
- Properties
  - ID
  - InsertDt
  - InsertStfpID
  - Part_ID
  - RaceCd
- Methods
  - New



ParticipantReferral
Class
→ WICBusinessObject

- Fields
  - _CaptureDt
  - _FollowupCd
  - _ID
  - _InsertDt
  - _InsertStfpID
  - _ModifyDt
  - _ModifyStfpID
  - _NoSignatureReasonCd
  - _Org_ID
  - _Part_ID
  - _RecordedDt
  - _SI_ID
  - _SPP_ID
  - _TypeCd
- Properties
  - CaptureDt
  - FollowupCd
  - ID
  - InsertDt
  - InsertStfpID
  - ModifyDt
  - ModifyStfpID
  - NoSignatureReasonCd
  - Org_ID
  - Part_ID
  - RecordedDt
  - SI_ID
  - SPP_ID
  - TypeCd
- Methods
  - New

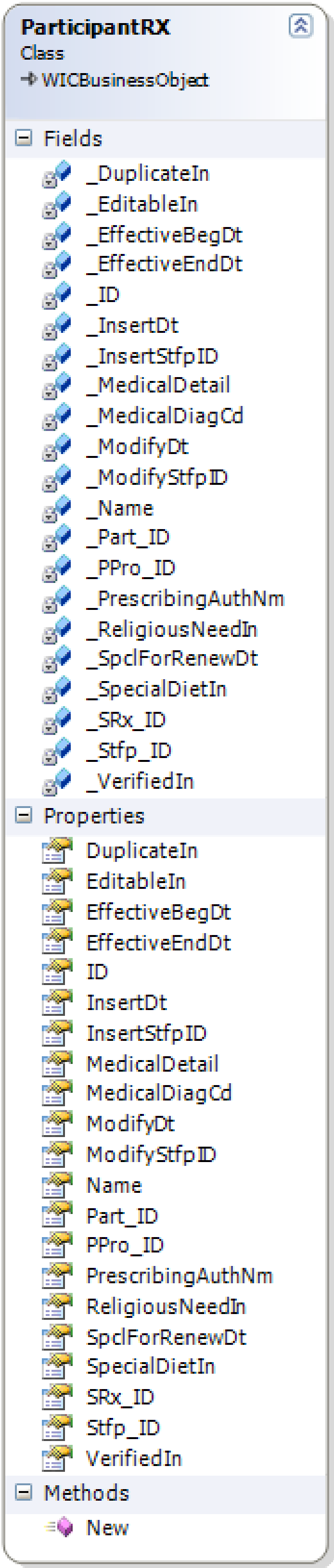


ParticipantRetri...
Class
→ WICBusinessObject

Fields
- _EndDt
- _FFClinicID
- _FFFamilyID
- _FFLocalAgencyID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Part_ID
- _PGFullName

Properties
- EndDt
- FFClinicID
- FFFamilyID
- FFLocalAgencyID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Part_ID
- PGFullName

Methods
- New

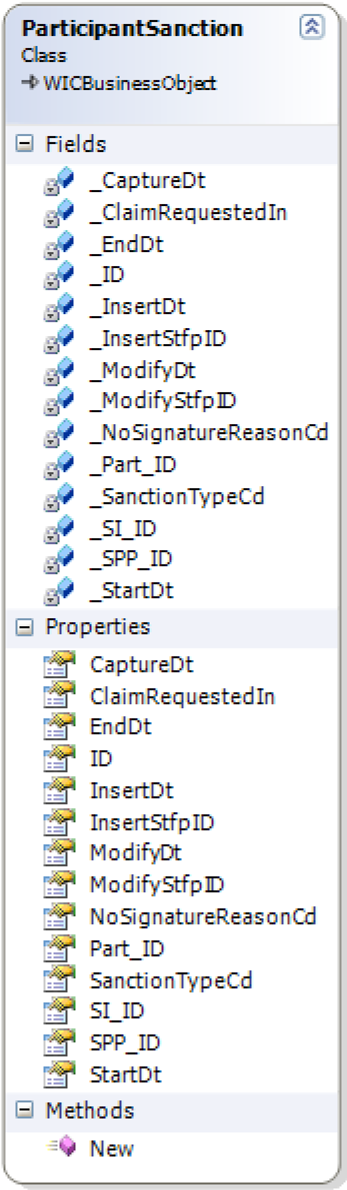


ParticipantRX
Class
→ WICBusinessObject

Fields
- _DuplicateIn
- _EditableIn
- _EffectiveBegDt
- _EffectiveEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _MedicalDetail
- _MedicalDiagCd
- _ModifyDt
- _ModifyStfpID
- _Name
- _Part_ID
- _PPro_ID
- _PrescribingAuthNm
- _ReligiousNeedIn
- _SpclForRenewDt
- _SpecialDietIn
- _SRx_ID
- _Stfp_ID
- _VerifiedIn

Properties
- DuplicateIn
- EditableIn
- EffectiveBegDt
- EffectiveEndDt
- ID
- InsertDt
- InsertStfpID
- MedicalDetail
- MedicalDiagCd
- ModifyDt
- ModifyStfpID
- Name
- Part_ID
- PPro_ID
- PrescribingAuthNm
- ReligiousNeedIn
- SpclForRenewDt
- SpecialDietIn
- SRx_ID
- Stfp_ID
- VerifiedIn

Methods
- New

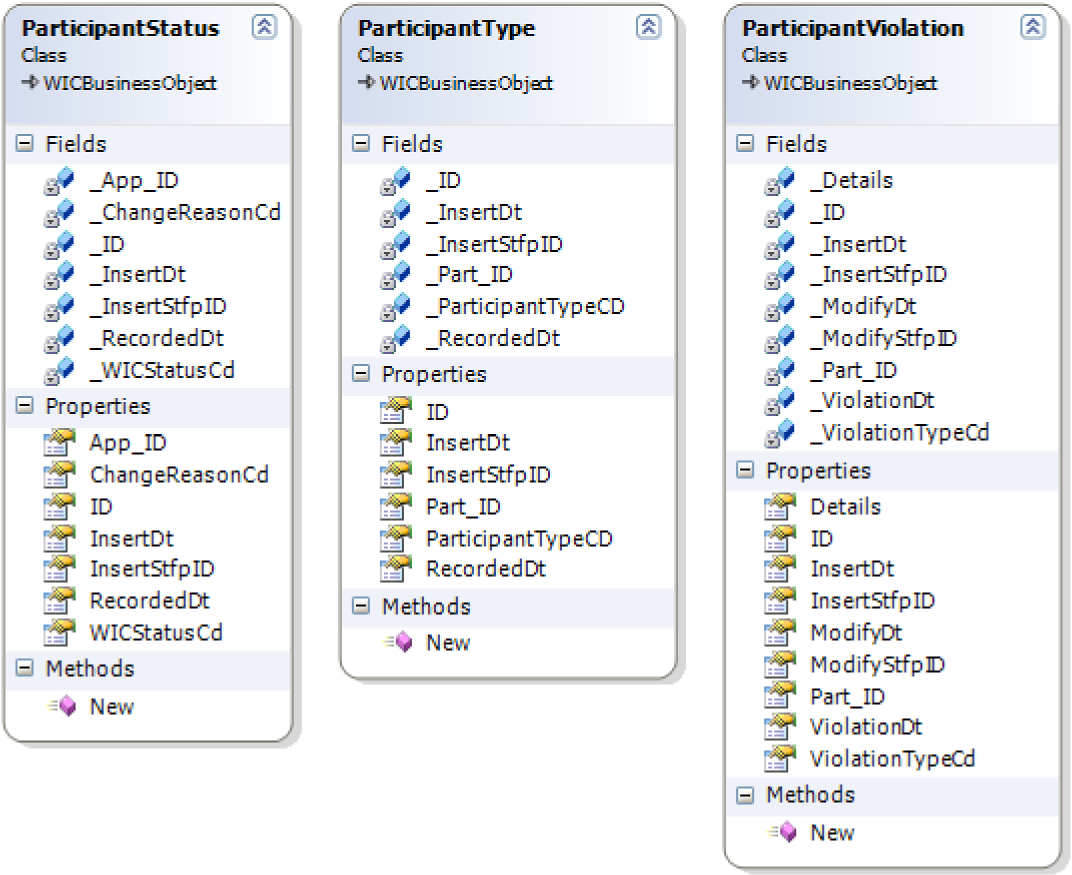


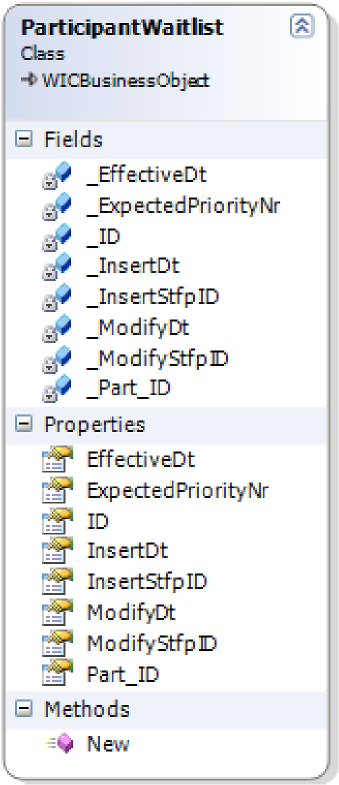
ParticipantSanction
Class
→ WICBusinessObject

Fields
- _CaptureDt
- _ClaimRequestedIn
- _EndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NoSignatureReasonCd
- _Part_ID
- _SanctionTypeCd
- _SI_ID
- _SPP_ID
- _StartDt

Properties
- CaptureDt
- ClaimRequestedIn
- EndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NoSignatureReasonCd
- Part_ID
- SanctionTypeCd
- SI_ID
- SPP_ID
- StartDt

Methods
- New


ParticipantStatus
Class
WICBusinessObject
Fields
_App_ID
_ChangeReasonCd
_ID
_InsertDt
_InsertStfpID
_RecordedDt
_WICStatusCd
Properties
App_ID
ChangeReasonCd
ID
InsertDt
InsertStfpID
RecordedDt
WICStatusCd
Methods
New
ParticipantType
Class
WICBusinessObject
Fields
_ID
_InsertDt
_InsertStfpID
_Part_ID
_ParticipantTypeCD
_RecordedDt
Properties
ID
InsertDt
InsertStfpID
Part_ID
ParticipantTypeCD
RecordedDt
Methods
New
ParticipantViolation
Class
WICBusinessObject
Fields
_Details
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_Part_ID
_ViolationDt
_ViolationTypeCd
Properties
Details
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Part_ID
ViolationDt
ViolationTypeCd
Methods
New

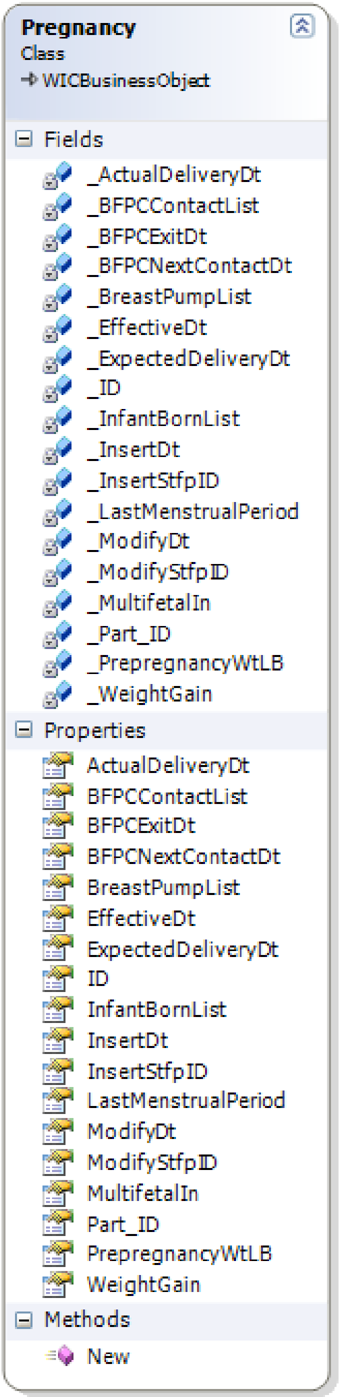


ParticipantWaitlist
Class
→ WICBusinessObject

Fields
- _EffectiveDt
- _ExpectedPriorityNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Part_ID

Properties
- EffectiveDt
- ExpectedPriorityNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Part_ID

Methods
- New

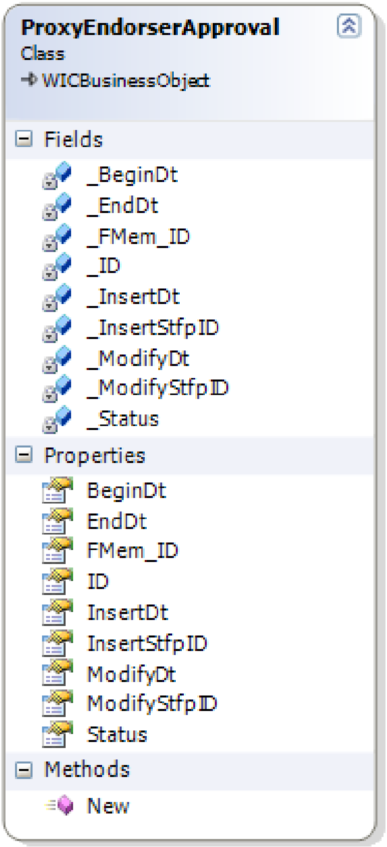


Pregnancy
Class
→ WICBusinessObject

Fields
- _ActualDeliveryDt
- _BFPCContactList
- _BFPCExitDt
- _BFPCNextContactDt
- _BreastPumpList
- _EffectiveDt
- _ExpectedDeliveryDt
- _ID
- _InfantBornList
- _InsertDt
- _InsertStfpID
- _LastMenstrualPeriod
- _ModifyDt
- _ModifyStfpID
- _MultifetalIn
- _Part_ID
- _PrepregnancyWtLB
- _WeightGain

Properties
- ActualDeliveryDt
- BFPCContactList
- BFPCExitDt
- BFPCNextContactDt
- BreastPumpList
- EffectiveDt
- ExpectedDeliveryDt
- ID
- InfantBornList
- InsertDt
- InsertStfpID
- LastMenstrualPeriod
- ModifyDt
- ModifyStfpID
- MultifetalIn
- Part_ID
- PrepregnancyWtLB
- WeightGain

Methods
- New



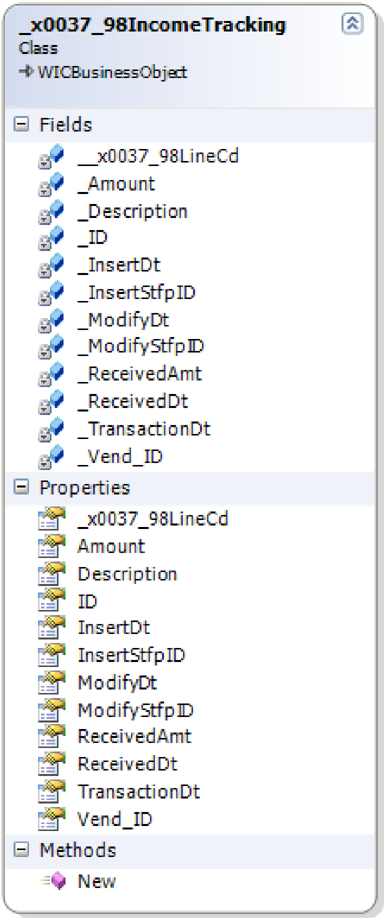
ProxyEndorserApproval
Class
→ WICBusinessObject

Fields
- _BeginDt
- _EndDt
- _FMem_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Status

Properties
- BeginDt
- EndDt
- FMem_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Status

Methods
- New

11.3 Finance

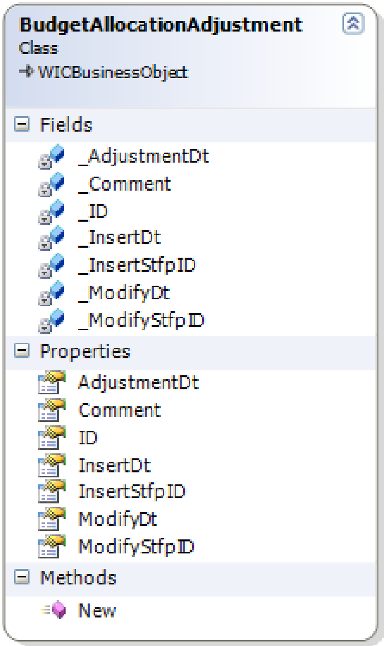


_x0037_98IncomeTracking
Class
WICBusinessObject

Fields
- __x0037_98LineCd
- _Amount
- _Description
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _ReceivedAmt
- _ReceivedDt
- _TransactionDt
- _Vend_ID

Properties
- _x0037_98LineCd
- Amount
- Description
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- ReceivedAmt
- ReceivedDt
- TransactionDt
- Vend_ID

Methods
- New

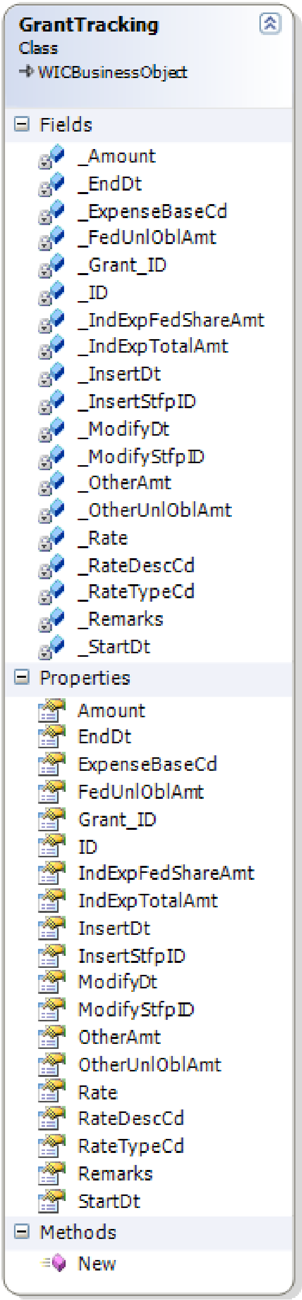


BudgetAllocationAdjustment
Class
WICBusinessObject

Fields
- _AdjustmentDt
- _Comment
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID

Properties
- AdjustmentDt
- Comment
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID

Methods
- New

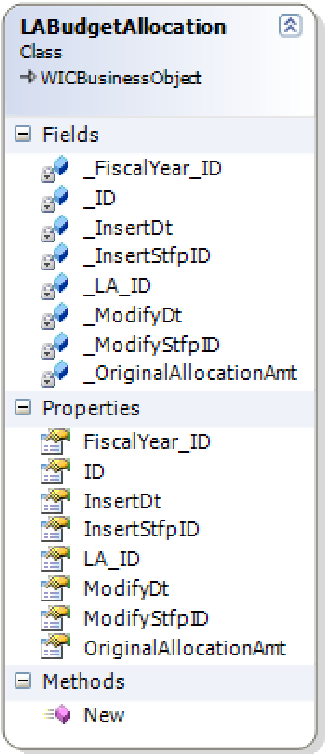


GrantTracking
Class
WICBusinessObject

Fields
- _Amount
- _EndDt
- _ExpenseBaseCd
- _FedUnlOblAmt
- _Grant_ID
- _ID
- _IndExpFedShareAmt
- _IndExpTotalAmt
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OtherAmt
- _OtherUnlOblAmt
- _Rate
- _RateDescCd
- _RateTypeCd
- _Remarks
- _StartDt

Properties
- Amount
- EndDt
- ExpenseBaseCd
- FedUnlOblAmt
- Grant_ID
- ID
- IndExpFedShareAmt
- IndExpTotalAmt
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OtherAmt
- OtherUnlOblAmt
- Rate
- RateDescCd
- RateTypeCd
- Remarks
- StartDt

Methods
- New

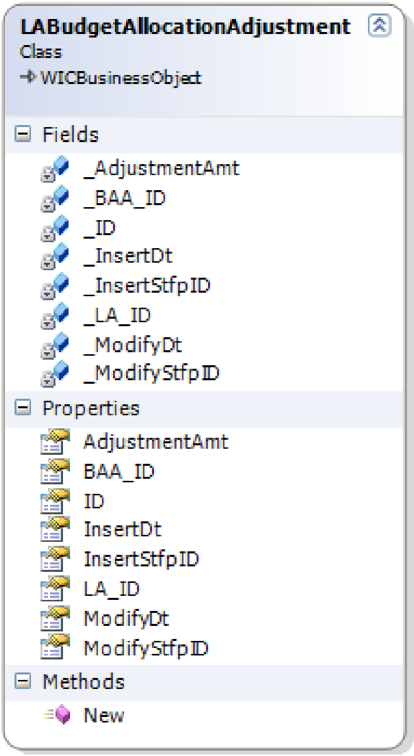


LABudgetAllocation
Class
→ WICBusinessObject

Fields
- _FiscalYear_ID
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _OriginalAllocationAmt

Properties
- FiscalYear_ID
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- OriginalAllocationAmt

Methods
- New

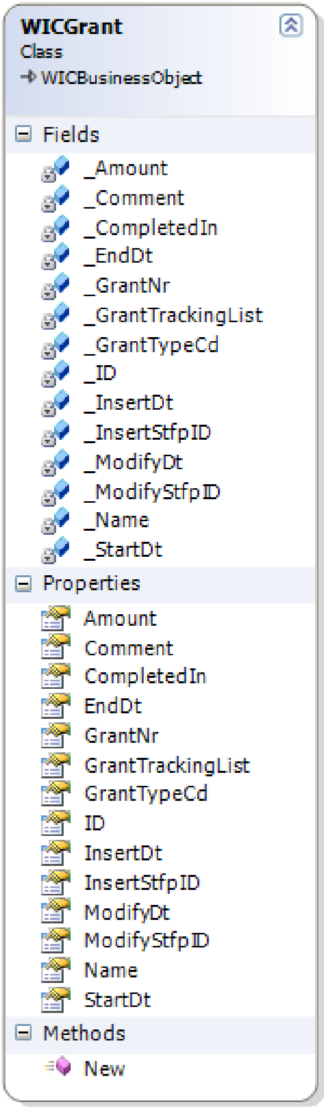


LABudgetAllocationAdjustment
Class
→ WICBusinessObject

Fields
- _AdjustmentAmt
- _BAA_ID
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID

Properties
- AdjustmentAmt
- BAA_ID
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID

Methods
- New



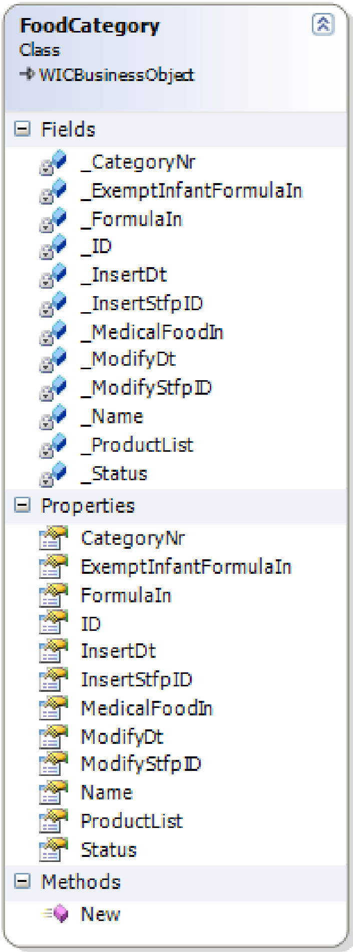
WICGrant
Class
→ WICBusinessObject

Fields
- _Amount
- _Comment
- _CompletedIn
- _EndDt
- _GrantNr
- _GrantTrackingList
- _GrantTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _StartDt

Properties
- Amount
- Comment
- CompletedIn
- EndDt
- GrantNr
- GrantTrackingList
- GrantTypeCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- StartDt

Methods
- New

11.4 Food Management



FoodCategory
Class
→ WICBusinessObject

Fields

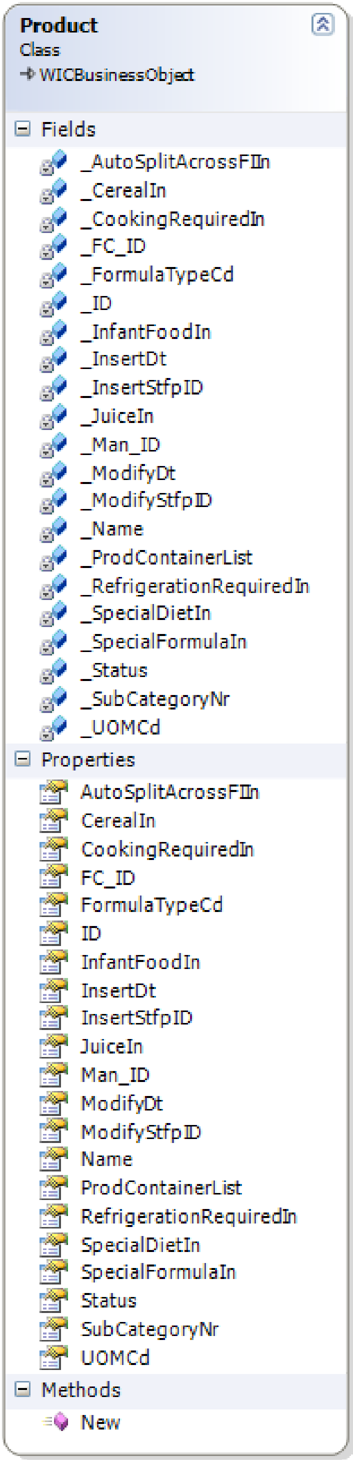
- _CategoryNr
- _ExemptInfantFormulaIn
- _FormulaIn
- _ID
- _InsertDt
- _InsertStfpID
- _MedicalFoodIn
- _ModifyDt
- _ModifyStfpID
- _Name
- _ProductList
- _Status

Properties

- CategoryNr
- ExemptInfantFormulaIn
- FormulaIn
- ID
- InsertDt
- InsertStfpID
- MedicalFoodIn
- ModifyDt
- ModifyStfpID
- Name
- ProductList
- Status

Methods

- New



Product
Class
→ WICBusinessObject

Fields

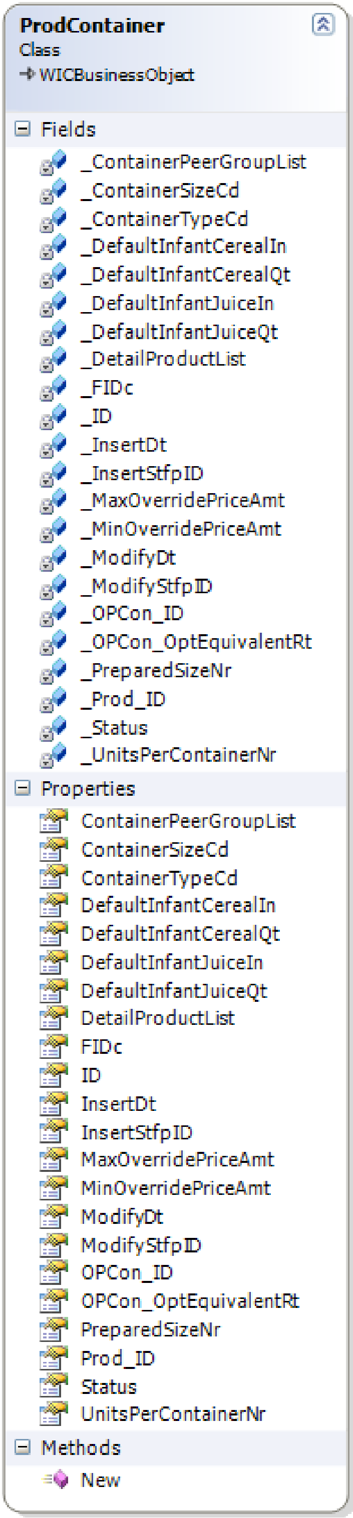
- _AutoSplitAcrossFIIn
- _CerealIn
- _CookingRequiredIn
- _FC_ID
- _FormulaTypeCd
- _ID
- _InfantFoodIn
- _InsertDt
- _InsertStfpID
- _JuiceIn
- _Man_ID
- _ModifyDt
- _ModifyStfpID
- _Name
- _ProdContainerList
- _RefrigerationRequiredIn
- _SpecialDietIn
- _SpecialFormulaIn
- _Status
- _SubCategoryNr
- _UOMCd

Properties

- AutoSplitAcrossFIIn
- CerealIn
- CookingRequiredIn
- FC_ID
- FormulaTypeCd
- ID
- InfantFoodIn
- InsertDt
- InsertStfpID
- JuiceIn
- Man_ID
- ModifyDt
- ModifyStfpID
- Name
- ProdContainerList
- RefrigerationRequiredIn
- SpecialDietIn
- SpecialFormulaIn
- Status
- SubCategoryNr
- UOMCd

Methods

- New



ProdContainer
Class
→ WICBusinessObject

Fields

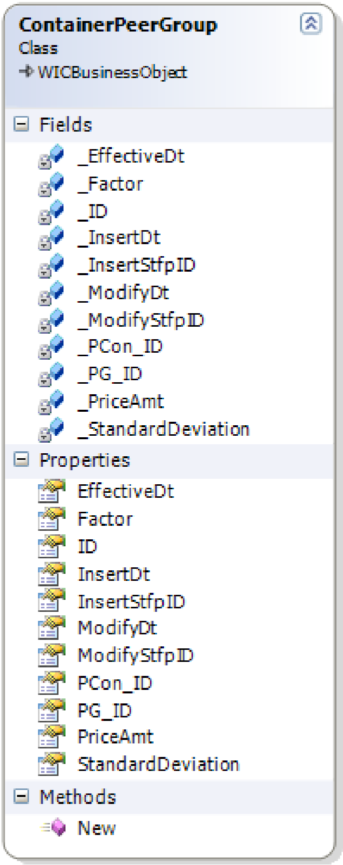
- _ContainerPeerGroupList
- _ContainerSizeCd
- _ContainerTypeCd
- _DefaultInfantCerealIn
- _DefaultInfantCerealQt
- _DefaultInfantJuiceIn
- _DefaultInfantJuiceQt
- _DetailProductList
- _FIDc
- _ID
- _InsertDt
- _InsertStfpID
- _MaxOverridePriceAmt
- _MinOverridePriceAmt
- _ModifyDt
- _ModifyStfpID
- _OPCon_ID
- _OPCon_OptEquivalentRt
- _PreparedSizeNr
- _Prod_ID
- _Status
- _UnitsPerContainerNr

Properties

- ContainerPeerGroupList
- ContainerSizeCd
- ContainerTypeCd
- DefaultInfantCerealIn
- DefaultInfantCerealQt
- DefaultInfantJuiceIn
- DefaultInfantJuiceQt
- DetailProductList
- FIDc
- ID
- InsertDt
- InsertStfpID
- MaxOverridePriceAmt
- MinOverridePriceAmt
- ModifyDt
- ModifyStfpID
- OPCon_ID
- OPCon_OptEquivalentRt
- PreparedSizeNr
- Prod_ID
- Status
- UnitsPerContainerNr

Methods

- New

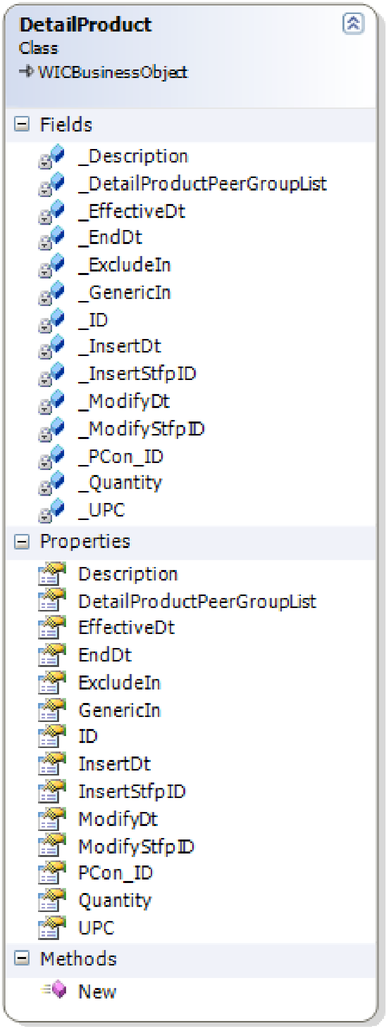


ContainerPeerGroup
Class
→ WICBusinessObject

Fields
- _EffectiveDt
- _Factor
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _PCon_ID
- _PG_ID
- _PriceAmt
- _StandardDeviation

Properties
- EffectiveDt
- Factor
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- PCon_ID
- PG_ID
- PriceAmt
- StandardDeviation

Methods
- New

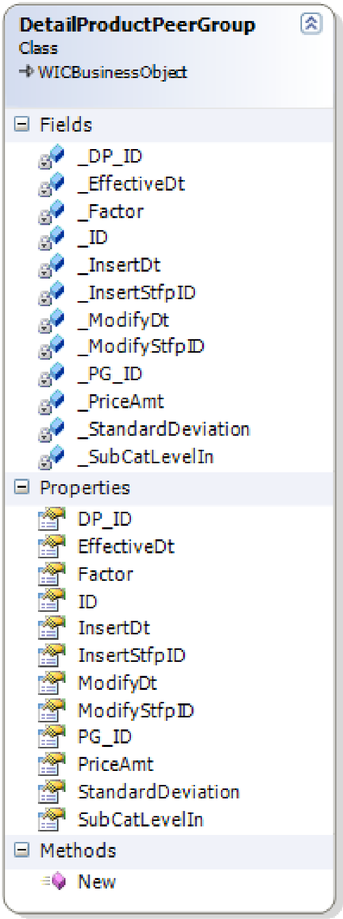


DetailProduct
Class
→ WICBusinessObject

Fields
- _Description
- _DetailProductPeerGroupList
- _EffectiveDt
- _EndDt
- _ExcludeIn
- _GenericIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _PCon_ID
- _Quantity
- _UPC

Properties
- Description
- DetailProductPeerGroupList
- EffectiveDt
- EndDt
- ExcludeIn
- GenericIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- PCon_ID
- Quantity
- UPC

Methods
- New

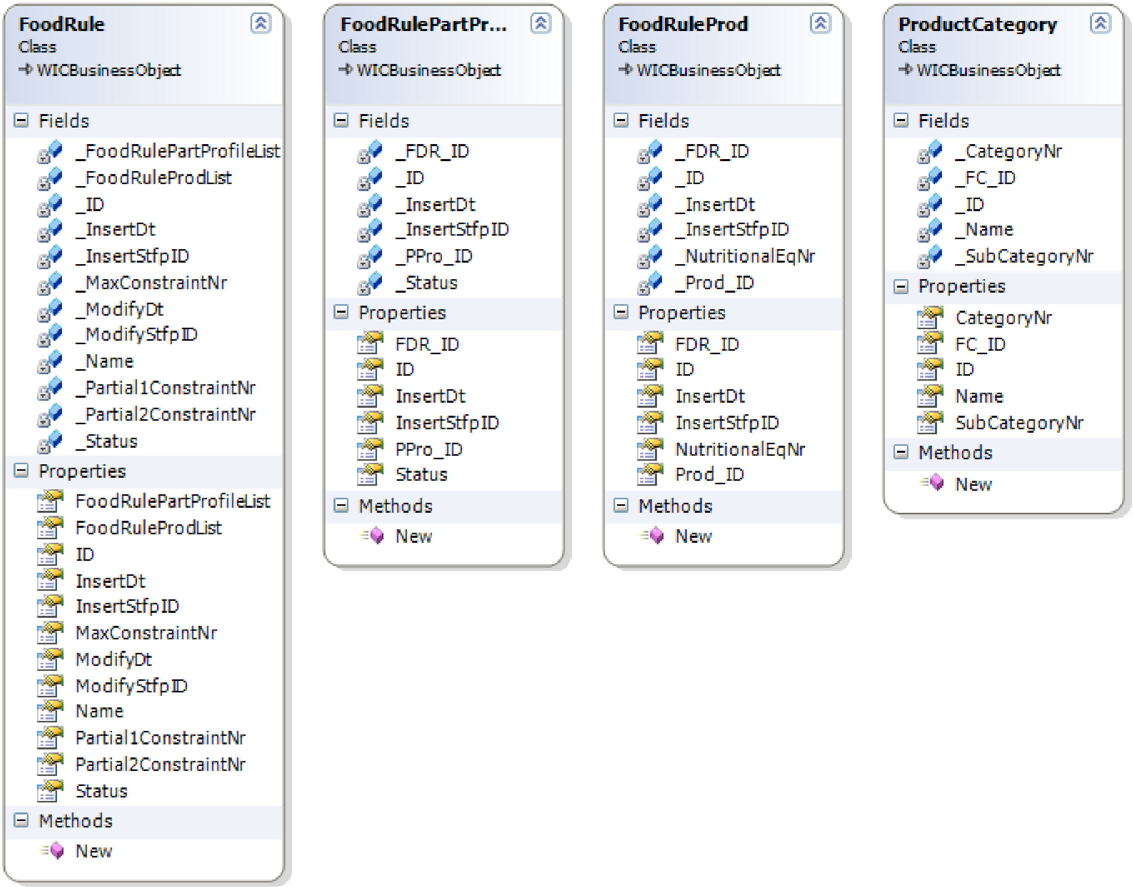


DetailProductPeerGroup
Class
→ WICBusinessObject

Fields
- _DP_ID
- _EffectiveDt
- _Factor
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _PG_ID
- _PriceAmt
- _StandardDeviation
- _SubCatLevelIn

Properties
- DP_ID
- EffectiveDt
- Factor
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- PG_ID
- PriceAmt
- StandardDeviation
- SubCatLevelIn

Methods
- New



FoodRule
Class
→ WICBusinessObject

Fields
- _FoodRulePartProfileList
- _FoodRuleProdList
- _ID
- _InsertDt
- _InsertStfpID
- _MaxConstraintNr
- _ModifyDt
- _ModifyStfpID
- _Name
- _Partial1ConstraintNr
- _Partial2ConstraintNr
- _Status

Properties
- FoodRulePartProfileList
- FoodRuleProdList
- ID
- InsertDt
- InsertStfpID
- MaxConstraintNr
- ModifyDt
- ModifyStfpID
- Name
- Partial1ConstraintNr
- Partial2ConstraintNr
- Status

Methods
- New

FoodRulePartPr...
Class
→ WICBusinessObject

Fields
- _FDR_ID
- _ID
- _InsertDt
- _InsertStfpID
- _PPro_ID
- _Status

Properties
- FDR_ID
- ID
- InsertDt
- InsertStfpID
- PPro_ID
- Status

Methods
- New

FoodRuleProd
Class
→ WICBusinessObject

Fields
- _FDR_ID
- _ID
- _InsertDt
- _InsertStfpID
- _NutritionalEqNr
- _Prod_ID

Properties
- FDR_ID
- ID
- InsertDt
- InsertStfpID
- NutritionalEqNr
- Prod_ID

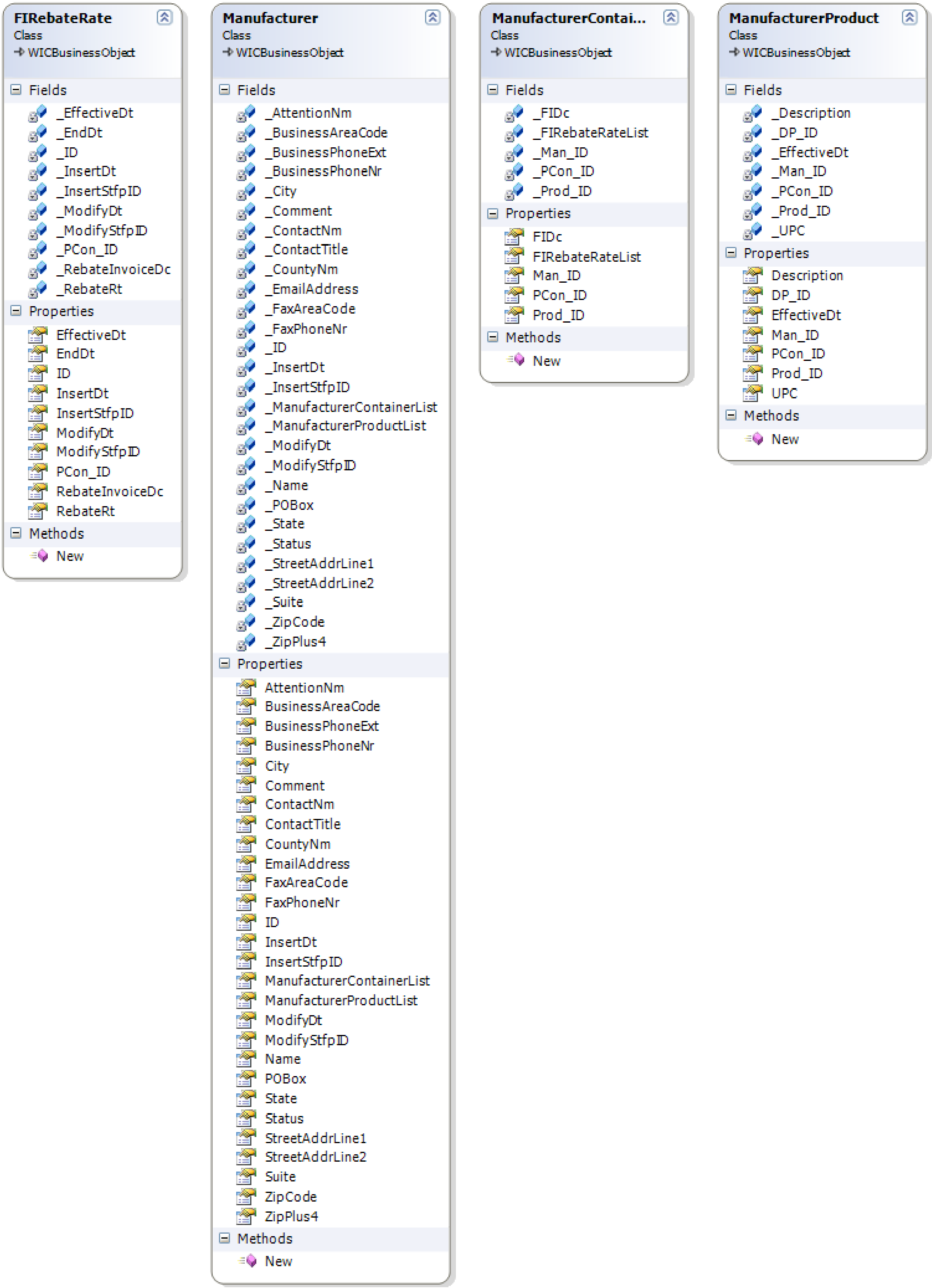
Methods
- New

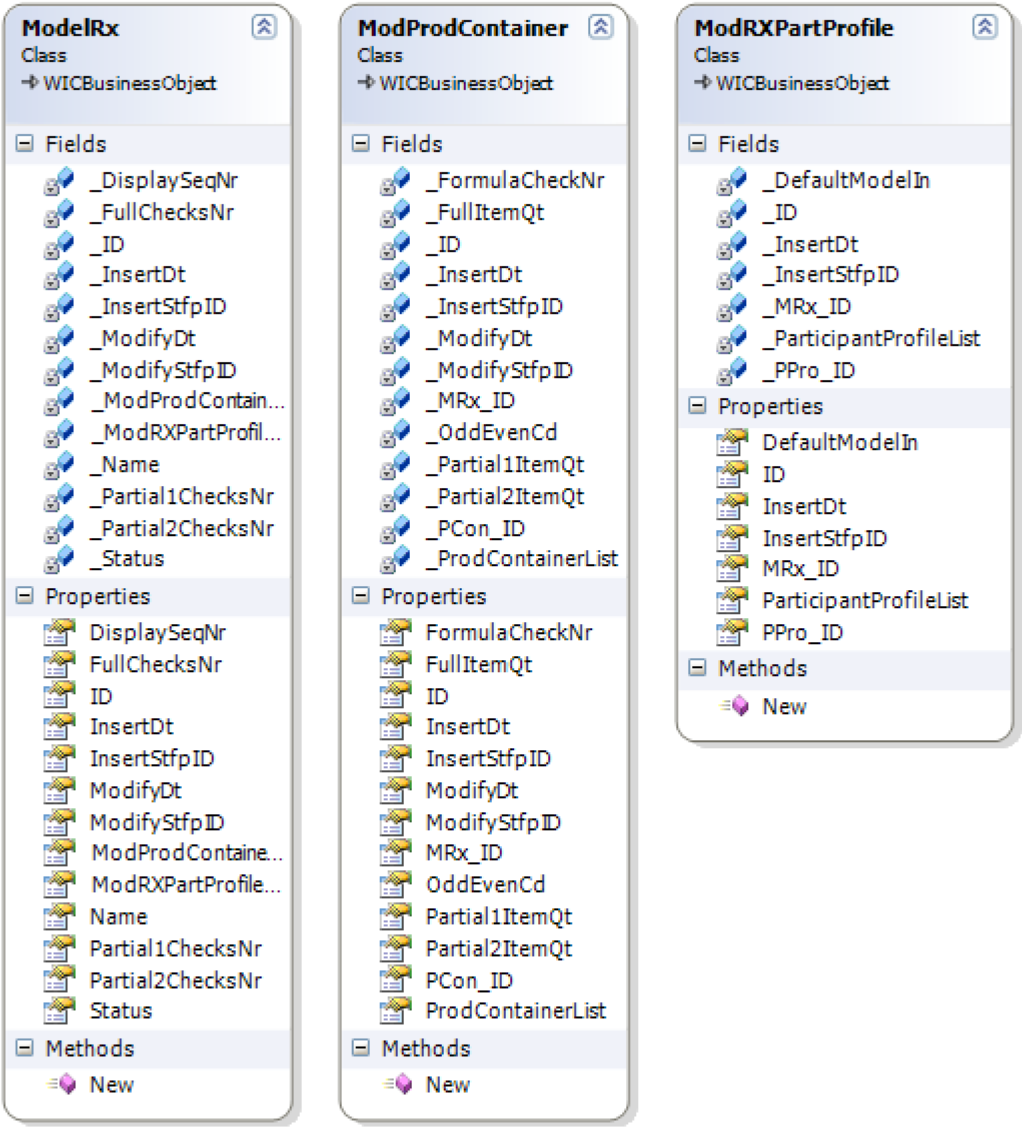
ProductCategory
Class
→ WICBusinessObject

Fields
- _CategoryNr
- _FC_ID
- _ID
- _Name
- _SubCategoryNr

Properties
- CategoryNr
- FC_ID
- ID
- Name
- SubCategoryNr

Methods
- New


FIRebateRate
Class
WICBusinessObject
Fields
_EffectiveDt
_EndDt
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_PCon_ID
_RebateInvoiceDc
_RebateRt
Properties
EffectiveDt
EndDt
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
PCon_ID
RebateInvoiceDc
RebateRt
Methods
New
Manufacturer
Class
WICBusinessObject
Fields
_AttentionNm
_BusinessAreaCode
_BusinessPhoneExt
_BusinessPhoneNr
_City
_Comment
_ContactNm
_ContactTitle
_CountyNm
_EmailAddress
_FaxAreaCode
_FaxPhoneNr
_ID
_InsertDt
_InsertStfpID
_ManufacturerContainerList
_ManufacturerProductList
_ModifyDt
_ModifyStfpID
_Name
_POBox
_State
_Status
_StreetAddrLine1
_StreetAddrLine2
_Suite
_ZipCode
_ZipPlus4
Properties
AttentionNm
BusinessAreaCode
BusinessPhoneExt
BusinessPhoneNr
City
Comment
ContactNm
ContactTitle
CountyNm
EmailAddress
FaxAreaCode
FaxPhoneNr
ID
InsertDt
InsertStfpID
ManufacturerContainerList
ManufacturerProductList
ModifyDt
ModifyStfpID
Name
POBox
State
Status
StreetAddrLine1
StreetAddrLine2
Suite
ZipCode
ZipPlus4
Methods
New
ManufacturerContai...
Class
WICBusinessObject
Fields
_FIDc
_FIRebateRateList
_Man_ID
_PCon_ID
_Prod_ID
Properties
FIDc
FIRebateRateList
Man_ID
PCon_ID
Prod_ID
Methods
New
ManufacturerProduct
Class
WICBusinessObject
Fields
_Description
_DP_ID
_EffectiveDt
_Man_ID
_PCon_ID
_Prod_ID
_UPC
Properties
Description
DP_ID
EffectiveDt
Man_ID
PCon_ID
Prod_ID
UPC
Methods
New



ModelRx
Class
→ WICBusinessObject

Fields
- _DisplaySeqNr
- _FullChecksNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _ModProdContain...
- _ModRXPartProfil...
- _Name
- _Partial1ChecksNr
- _Partial2ChecksNr
- _Status

Properties
- DisplaySeqNr
- FullChecksNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- ModProdContaine...
- ModRXPartProfile...
- Name
- Partial1ChecksNr
- Partial2ChecksNr
- Status

Methods
- New

ModProdContainer
Class
→ WICBusinessObject

Fields
- _FormulaCheckNr
- _FullItemQt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _MRx_ID
- _OddEvenCd
- _Partial1ItemQt
- _Partial2ItemQt
- _PCon_ID
- _ProdContainerList

Properties
- FormulaCheckNr
- FullItemQt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- MRx_ID
- OddEvenCd
- Partial1ItemQt
- Partial2ItemQt
- PCon_ID
- ProdContainerList

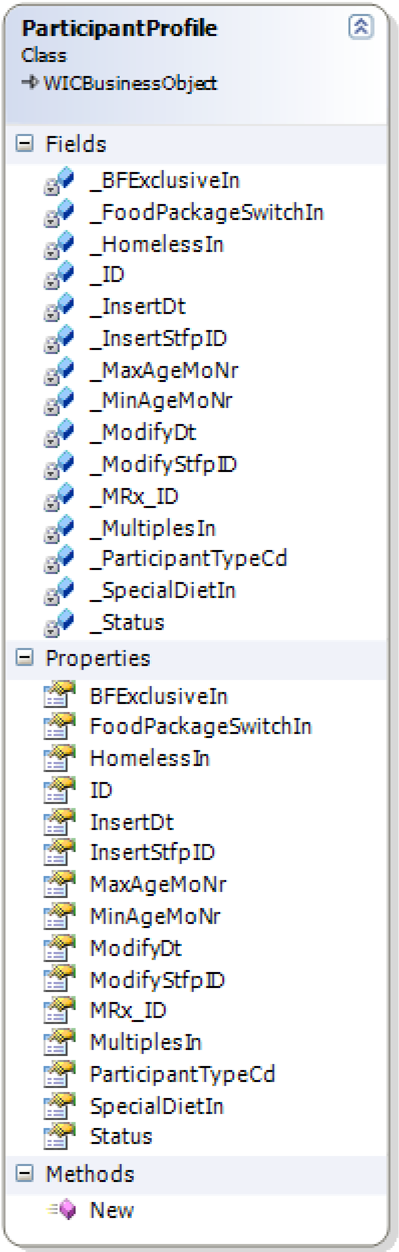
Methods
- New

ModRXPartProfile
Class
→ WICBusinessObject

Fields
- _DefaultModelIn
- _ID
- _InsertDt
- _InsertStfpID
- _MRx_ID
- _ParticipantProfileList
- _PPro_ID

Properties
- DefaultModelIn
- ID
- InsertDt
- InsertStfpID
- MRx_ID
- ParticipantProfileList
- PPro_ID

Methods
- New



ParticipantProfile
Class
→ WICBusinessObject

Fields

_BFExclusiveIn
_FoodPackageSwitchIn
_HomelessIn
_ID
_InsertDt
_InsertStfpID
_MaxAgeMoNr
_MinAgeMoNr
_ModifyDt
_ModifyStfpID
_MRx_ID
_MultiplesIn
_ParticipantTypeCd
_SpecialDietIn
_Status

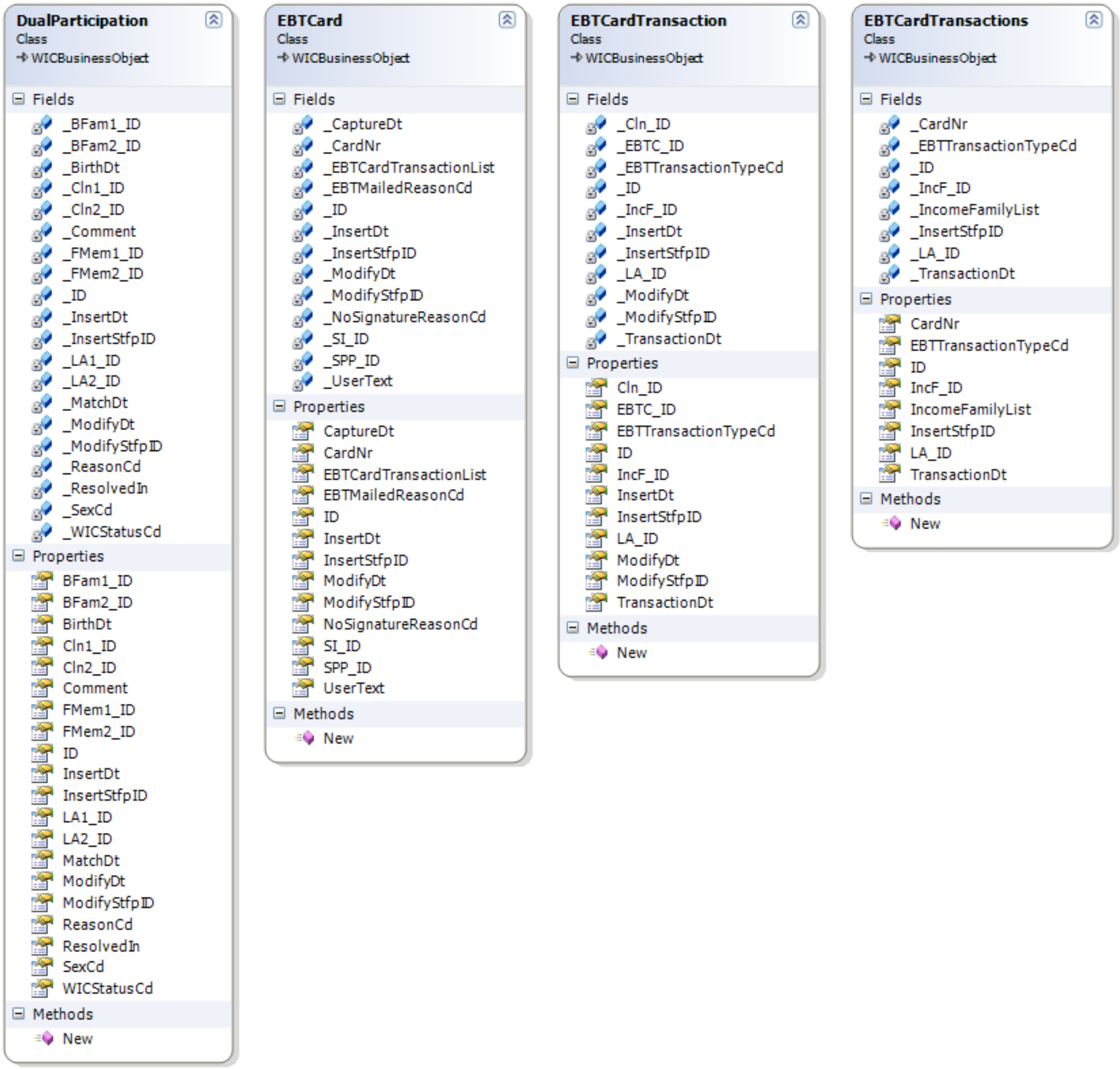
Properties

BFExclusiveIn
FoodPackageSwitchIn
HomelessIn
ID
InsertDt
InsertStfpID
MaxAgeMoNr
MinAgeMoNr
ModifyDt
ModifyStfpID
MRx_ID
MultiplesIn
ParticipantTypeCd
SpecialDietIn
Status

Methods

New

11.5 Operations



DualParticipation
Class
→ WICBusinessObject

Fields
- _BFam1_ID
- _BFam2_ID
- _BirthDt
- _Cln1_ID
- _Cln2_ID
- _Comment
- _FMem1_ID
- _FMem2_ID
- _ID
- _InsertDt
- _InsertStfpID
- _LA1_ID
- _LA2_ID
- _MatchDt
- _ModifyDt
- _ModifyStfpID
- _ReasonCd
- _ResolvedIn
- _SexCd
- _WICStatusCd

Properties
- BFam1_ID
- BFam2_ID
- BirthDt
- Cln1_ID
- Cln2_ID
- Comment
- FMem1_ID
- FMem2_ID
- ID
- InsertDt
- InsertStfpID
- LA1_ID
- LA2_ID
- MatchDt
- ModifyDt
- ModifyStfpID
- ReasonCd
- ResolvedIn
- SexCd
- WICStatusCd

Methods
- New

EBTCard
Class
→ WICBusinessObject

Fields
- _CaptureDt
- _CardNr
- _EBTCardTransactionList
- _EBTMailedReasonCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NoSignatureReasonCd
- _SI_ID
- _SPP_ID
- _UserText

Properties
- CaptureDt
- CardNr
- EBTCardTransactionList
- EBTMailedReasonCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NoSignatureReasonCd
- SI_ID
- SPP_ID
- UserText

Methods
- New

EBTCardTransaction
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _EBTC_ID
- _EBTTransactionTypeCd
- _ID
- _IncF_ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _TransactionDt

Properties
- Cln_ID
- EBTC_ID
- EBTTransactionTypeCd
- ID
- IncF_ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- TransactionDt

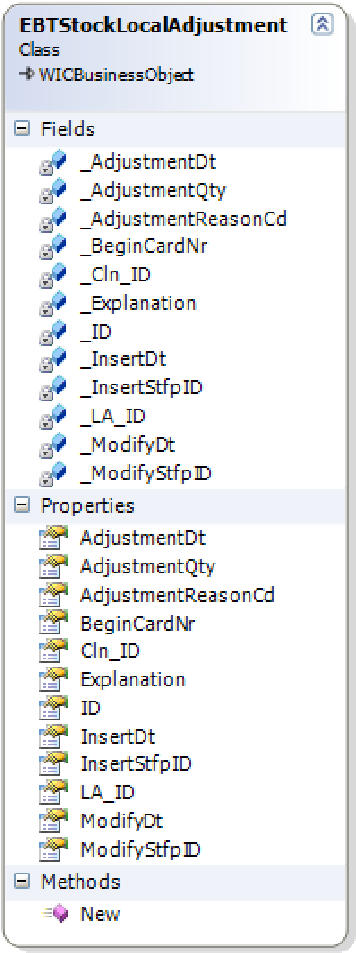
Methods
- New

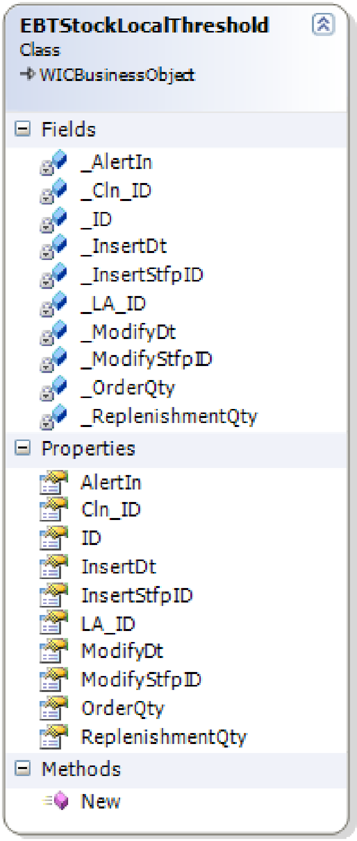
EBTCardTransactions
Class
→ WICBusinessObject

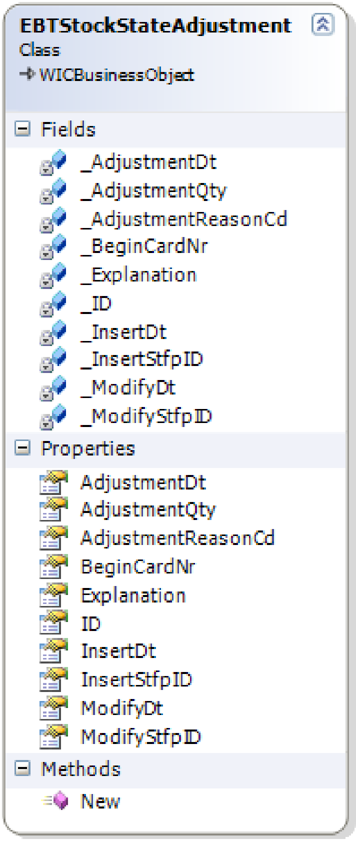
Fields
- _CardNr
- _EBTTransactionTypeCd
- _ID
- _IncF_ID
- _IncomeFamilyList
- _InsertStfpID
- _LA_ID
- _TransactionDt

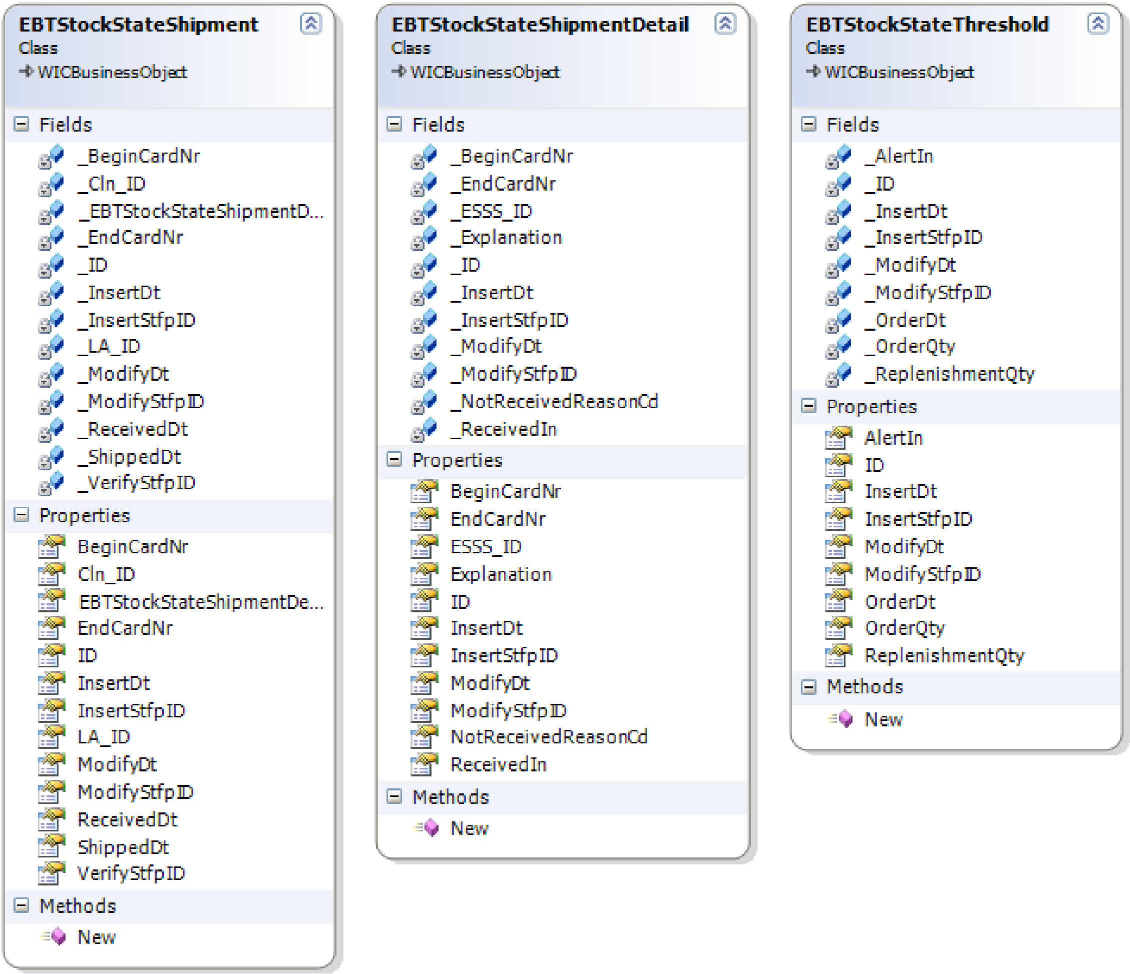
Properties
- CardNr
- EBTTransactionTypeCd
- ID
- IncF_ID
- IncomeFamilyList
- InsertStfpID
- LA_ID
- TransactionDt

Methods
- New


EBTStockLocalAdjustment
Class
WICBusinessObject
Fields
_AdjustmentDt
_AdjustmentQty
_AdjustmentReasonCd
_BeginCardNr
_Cln_ID
_Explanation
_ID
_InsertDt
_InsertStfpID
_LA_ID
_ModifyDt
_ModifyStfpID
Properties
AdjustmentDt
AdjustmentQty
AdjustmentReasonCd
BeginCardNr
Cln_ID
Explanation
ID
InsertDt
InsertStfpID
LA_ID
ModifyDt
ModifyStfpID
Methods
New


EBTStockLocalThreshold
Class
WICBusinessObject
Fields
_AlertIn
_Cln_ID
_ID
_InsertDt
_InsertStfpID
_LA_ID
_ModifyDt
_ModifyStfpID
_OrderQty
_ReplenishmentQty
Properties
AlertIn
Cln_ID
ID
InsertDt
InsertStfpID
LA_ID
ModifyDt
ModifyStfpID
OrderQty
ReplenishmentQty
Methods
New


EBTStockStateAdjustment
Class
WICBusinessObject
Fields
_AdjustmentDt
_AdjustmentQty
_AdjustmentReasonCd
_BeginCardNr
_Explanation
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
Properties
AdjustmentDt
AdjustmentQty
AdjustmentReasonCd
BeginCardNr
Explanation
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Methods
New



EBTStockStateShipment
Class
WICBusinessObject

Fields
- _BeginCardNr
- _Cln_ID
- _EBTStockStateShipmentD...
- _EndCardNr
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _ReceivedDt
- _ShippedDt
- _VerifyStfpID

Properties
- BeginCardNr
- Cln_ID
- EBTStockStateShipmentDe...
- EndCardNr
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- ReceivedDt
- ShippedDt
- VerifyStfpID

Methods
- New

EBTStockStateShipmentDetail
Class
WICBusinessObject

Fields
- _BeginCardNr
- _EndCardNr
- _ESSS_ID
- _Explanation
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NotReceivedReasonCd
- _ReceivedIn

Properties
- BeginCardNr
- EndCardNr
- ESSS_ID
- Explanation
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NotReceivedReasonCd
- ReceivedIn

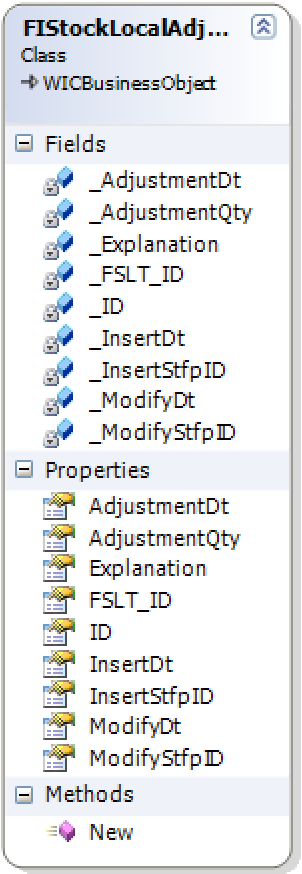
Methods
- New

EBTStockStateThreshold
Class
WICBusinessObject

Fields
- _AlertIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OrderDt
- _OrderQty
- _ReplenishmentQty

Properties
- AlertIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OrderDt
- OrderQty
- ReplenishmentQty

Methods
- New

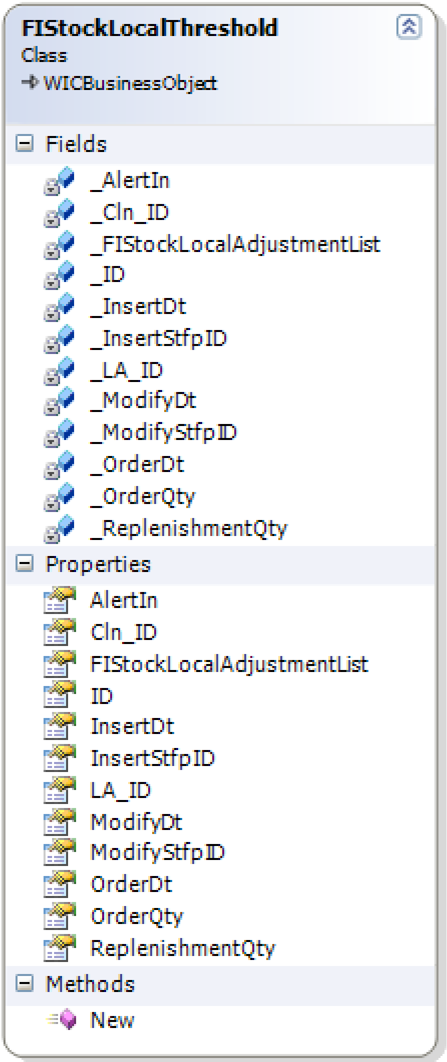


FIStockLocalAdj...
Class
→ WICBusinessObject

Fields
- _AdjustmentDt
- _AdjustmentQty
- _Explanation
- _FSLT_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID

Properties
- AdjustmentDt
- AdjustmentQty
- Explanation
- FSLT_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID

Methods
- New

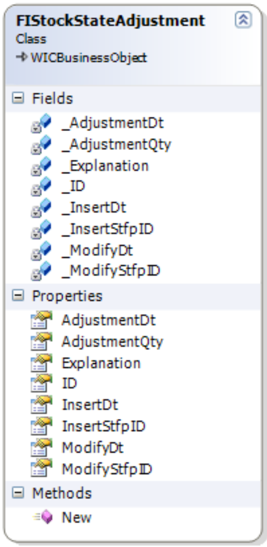


FIStockLocalThreshold
Class
→ WICBusinessObject

Fields
- _AlertIn
- _Cln_ID
- _FIStockLocalAdjustmentList
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _OrderDt
- _OrderQty
- _ReplenishmentQty

Properties
- AlertIn
- Cln_ID
- FIStockLocalAdjustmentList
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- OrderDt
- OrderQty
- ReplenishmentQty

Methods
- New

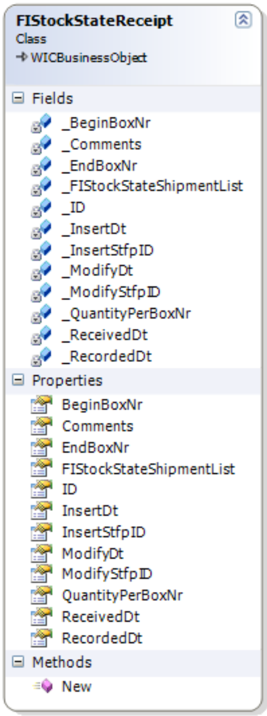


FIStockStateAdjustment
Class
WICBusinessObject

Fields
- _AdjustmentDt
- _AdjustmentQty
- _Explanation
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID

Properties
- AdjustmentDt
- AdjustmentQty
- Explanation
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID

Methods
- New

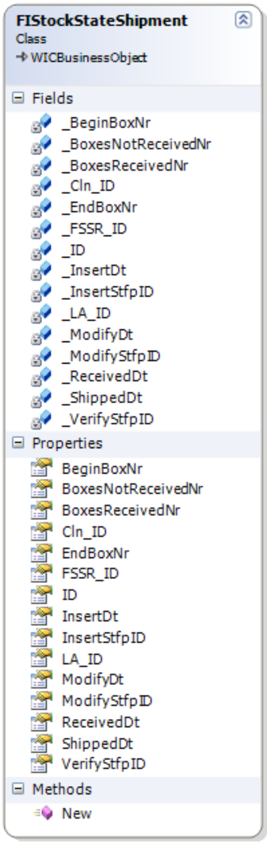


FIStockStateReceipt
Class
WICBusinessObject

Fields
- _BeginBoxNr
- _Comments
- _EndBoxNr
- _FIStockStateShipmentList
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _QuantityPerBoxNr
- _ReceivedDt
- _RecordedDt

Properties
- BeginBoxNr
- Comments
- EndBoxNr
- FIStockStateShipmentList
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- QuantityPerBoxNr
- ReceivedDt
- RecordedDt

Methods
- New

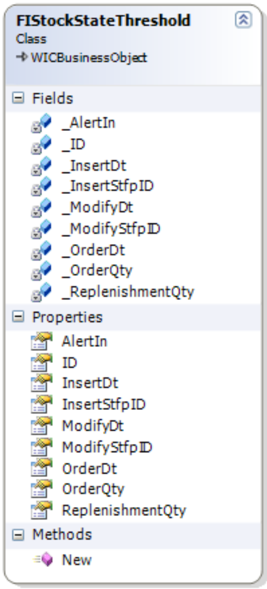


FIStockStateShipment
Class
WICBusinessObject

Fields
- _BeginBoxNr
- _BoxesNotReceivedNr
- _BoxesReceivedNr
- _Cln_ID
- _EndBoxNr
- _FSSR_ID
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _ReceivedDt
- _ShippedDt
- _VerifyStfpID

Properties
- BeginBoxNr
- BoxesNotReceivedNr
- BoxesReceivedNr
- Cln_ID
- EndBoxNr
- FSSR_ID
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- ReceivedDt
- ShippedDt
- VerifyStfpID

Methods
- New

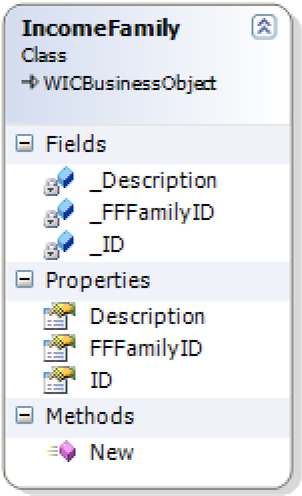


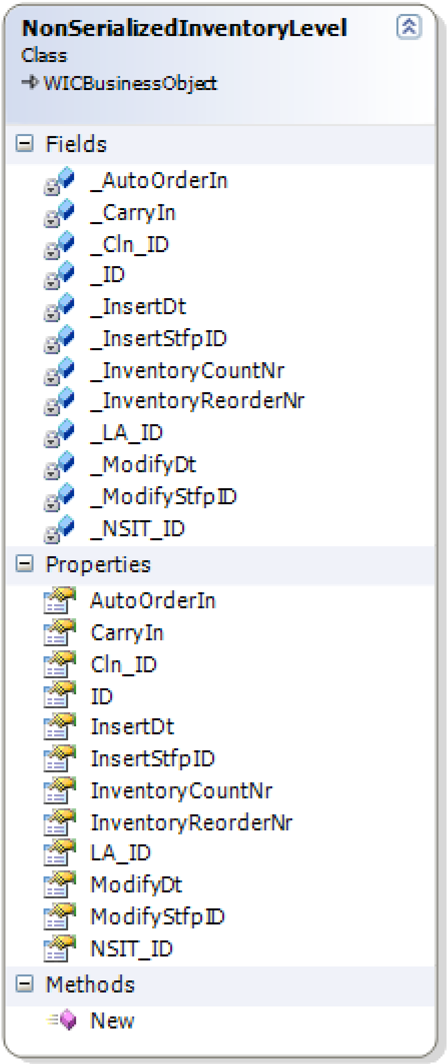
FIStockStateThreshold
Class
WICBusinessObject

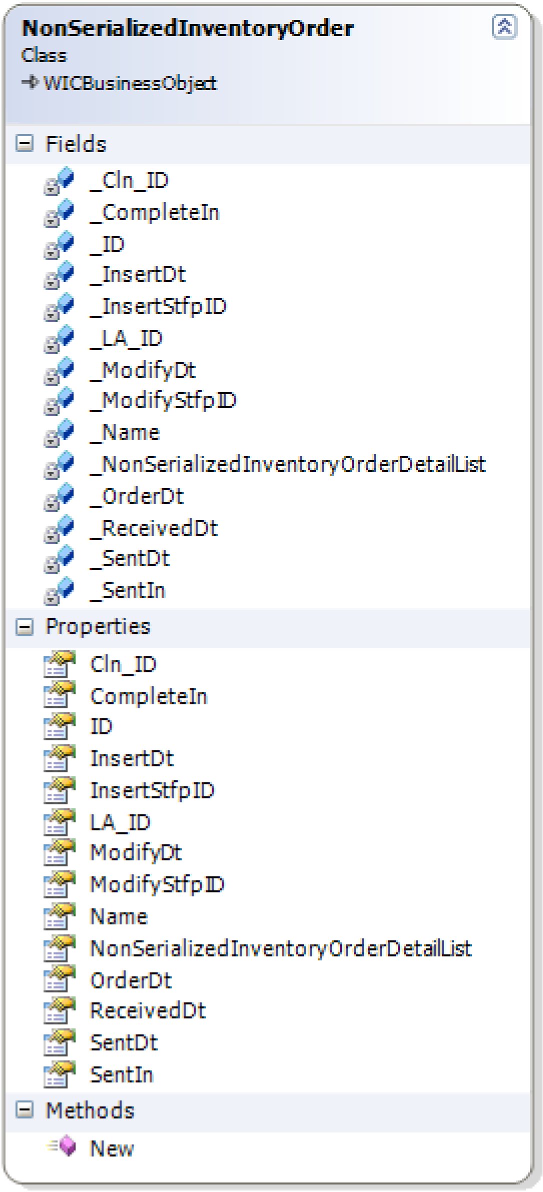
Fields
- _AlertIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OrderDt
- _OrderQty
- _ReplenishmentQty

Properties
- AlertIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OrderDt
- OrderQty
- ReplenishmentQty

Methods
- New


IncomeFamily
Class
WICBusinessObject
Fields
_Description
_FFFamilyID
_ID
Properties
Description
FFFamilyID
ID
Methods
New


NonSerializedInventoryLevel
Class
WICBusinessObject
Fields
_AutoOrderIn
_CarryIn
_Cln_ID
_ID
_InsertDt
_InsertStfpID
_InventoryCountNr
_InventoryReorderNr
_LA_ID
_ModifyDt
_ModifyStfpID
_NSIT_ID
Properties
AutoOrderIn
CarryIn
Cln_ID
ID
InsertDt
InsertStfpID
InventoryCountNr
InventoryReorderNr
LA_ID
ModifyDt
ModifyStfpID
NSIT_ID
Methods
New

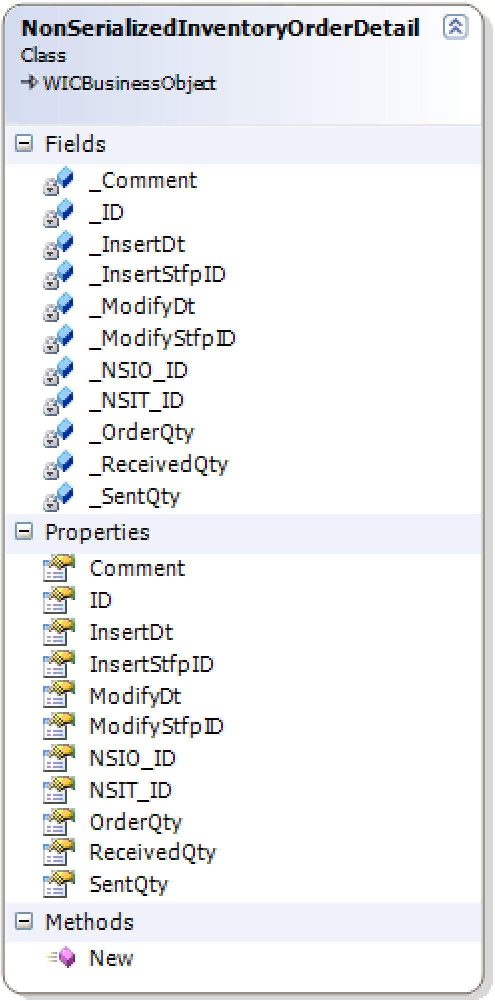


NonSerializedInventoryOrder
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _CompleteIn
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _Name
- _NonSerializedInventoryOrderDetailList
- _OrderDt
- _ReceivedDt
- _SentDt
- _SentIn

Properties
- Cln_ID
- CompleteIn
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- Name
- NonSerializedInventoryOrderDetailList
- OrderDt
- ReceivedDt
- SentDt
- SentIn

Methods
- New

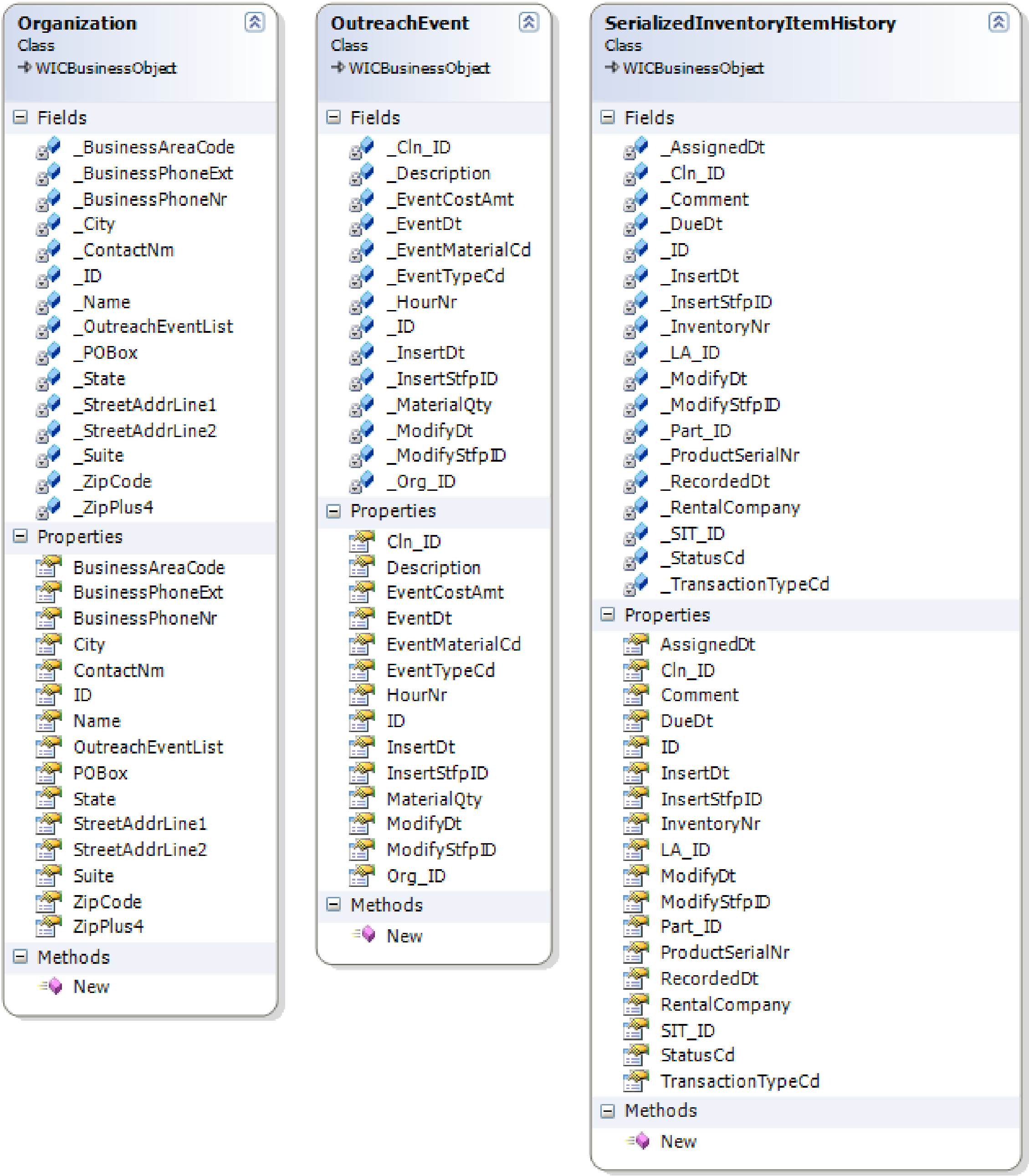


NonSerializedInventoryOrderDetail
Class
→ WICBusinessObject

Fields
- _Comment
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NSIO_ID
- _NSIT_ID
- _OrderQty
- _ReceivedQty
- _SentQty

Properties
- Comment
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NSIO_ID
- NSIT_ID
- OrderQty
- ReceivedQty
- SentQty

Methods
- New



Organization
Class
→ WICBusinessObject

Fields
- _BusinessAreaCode
- _BusinessPhoneExt
- _BusinessPhoneNr
- _City
- _ContactNm
- _ID
- _Name
- _OutreachEventList
- _POBox
- _State
- _StreetAddrLine1
- _StreetAddrLine2
- _Suite
- _ZipCode
- _ZipPlus4

Properties
- BusinessAreaCode
- BusinessPhoneExt
- BusinessPhoneNr
- City
- ContactNm
- ID
- Name
- OutreachEventList
- POBox
- State
- StreetAddrLine1
- StreetAddrLine2
- Suite
- ZipCode
- ZipPlus4

Methods
- New

OutreachEvent
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _Description
- _EventCostAmt
- _EventDt
- _EventMaterialCd
- _EventTypeCd
- _HourNr
- _ID
- _InsertDt
- _InsertStfpID
- _MaterialQty
- _ModifyDt
- _ModifyStfpID
- _Org_ID

Properties
- Cln_ID
- Description
- EventCostAmt
- EventDt
- EventMaterialCd
- EventTypeCd
- HourNr
- ID
- InsertDt
- InsertStfpID
- MaterialQty
- ModifyDt
- ModifyStfpID
- Org_ID

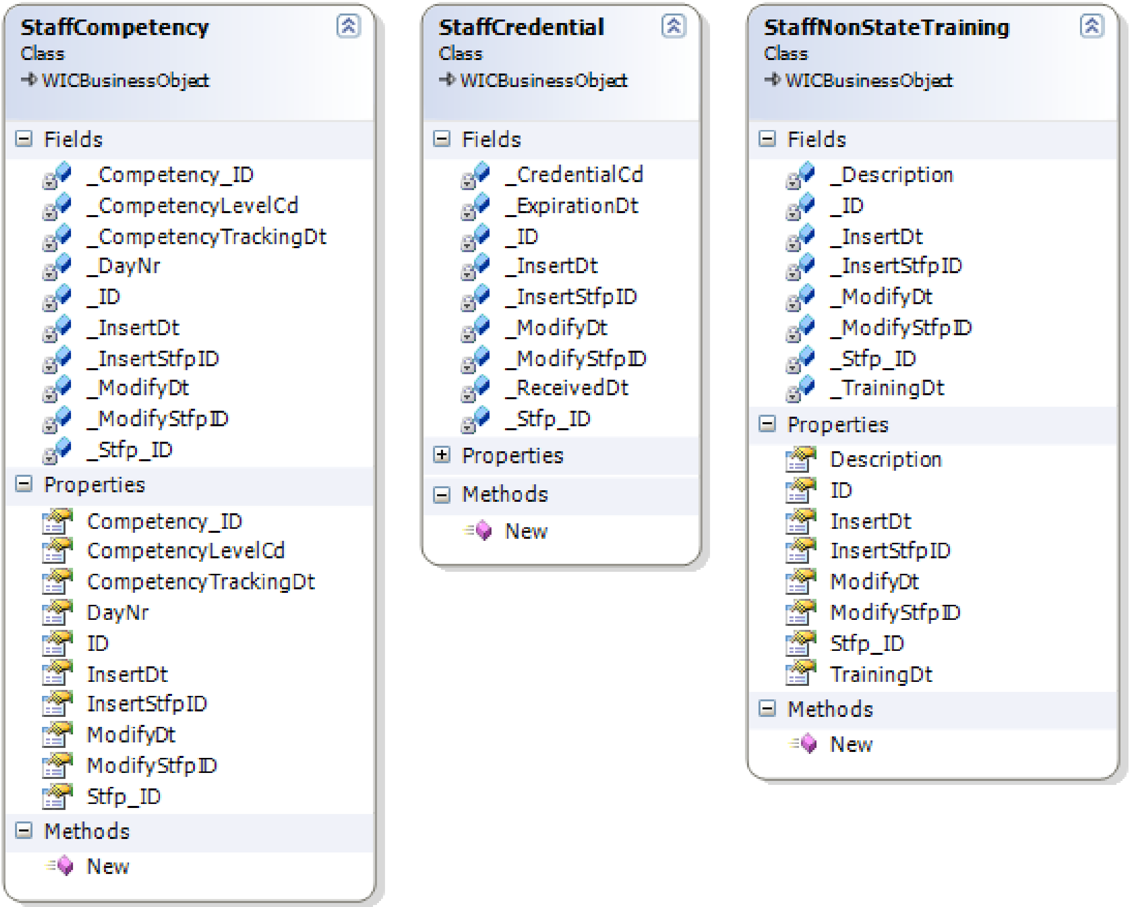
Methods
- New

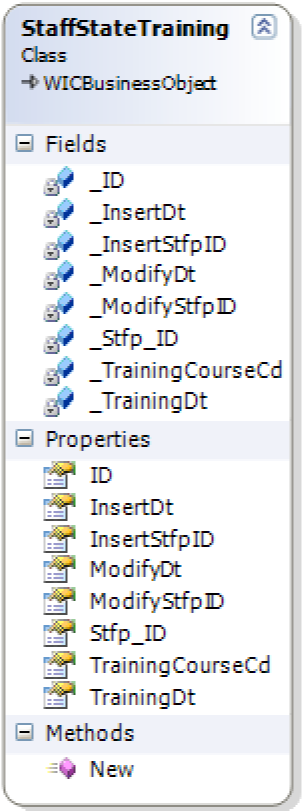
SerializedInventoryItemHistory
Class
→ WICBusinessObject

Fields
- _AssignedDt
- _Cln_ID
- _Comment
- _DueDt
- _ID
- _InsertDt
- _InsertStfpID
- _InventoryNr
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _Part_ID
- _ProductSerialNr
- _RecordedDt
- _RentalCompany
- _SIT_ID
- _StatusCd
- _TransactionTypeCd

Properties
- AssignedDt
- Cln_ID
- Comment
- DueDt
- ID
- InsertDt
- InsertStfpID
- InventoryNr
- LA_ID
- ModifyDt
- ModifyStfpID
- Part_ID
- ProductSerialNr
- RecordedDt
- RentalCompany
- SIT_ID
- StatusCd
- TransactionTypeCd

Methods
- New


StaffCompetency
Class
WICBusinessObject
Fields
_Competency_ID
_CompetencyLevelCd
_CompetencyTrackingDt
_DayNr
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_Stfp_ID
Properties
Competency_ID
CompetencyLevelCd
CompetencyTrackingDt
DayNr
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Stfp_ID
Methods
New
StaffCredential
Class
WICBusinessObject
Fields
_CredentialCd
_ExpirationDt
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_ReceivedDt
_Stfp_ID
Properties
Methods
New
StaffNonStateTraining
Class
WICBusinessObject
Fields
_Description
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_Stfp_ID
_TrainingDt
Properties
Description
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Stfp_ID
TrainingDt
Methods
New



StaffStateTraining
Class
→ WICBusinessObject

Fields

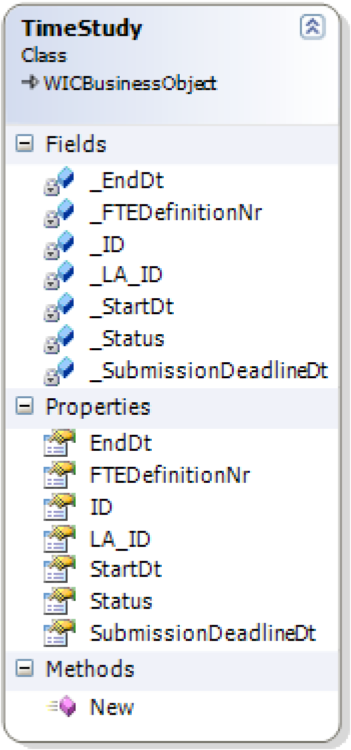
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Stfp_ID
- _TrainingCourseCd
- _TrainingDt

Properties

- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Stfp_ID
- TrainingCourseCd
- TrainingDt

Methods

- New



TimeStudy
Class
→ WICBusinessObject

Fields

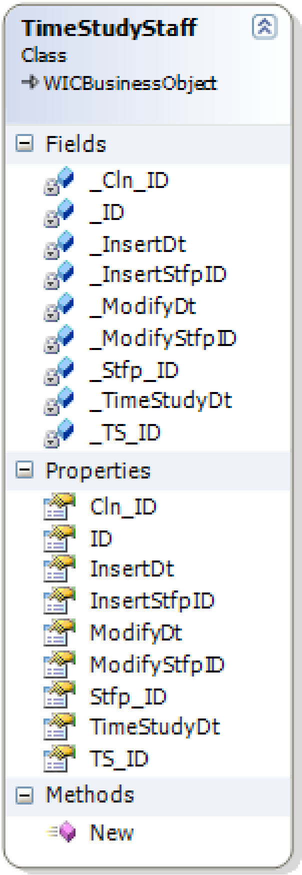
- _EndDt
- _FTEDefinitionNr
- _ID
- _LA_ID
- _StartDt
- _Status
- _SubmissionDeadlineDt

Properties

- EndDt
- FTEDefinitionNr
- ID
- LA_ID
- StartDt
- Status
- SubmissionDeadlineDt

Methods

- New



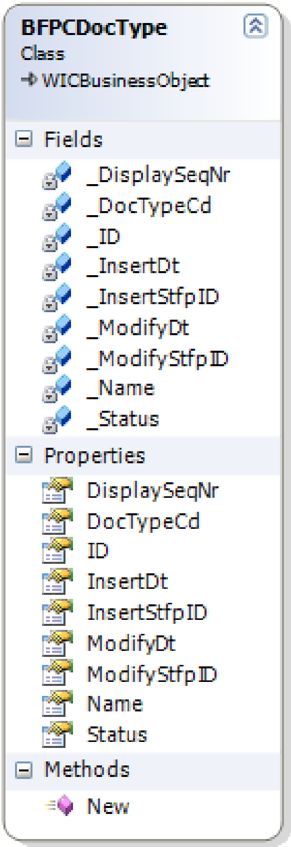
TimeStudyStaff
Class
WICBusinessObject

Fields
- _Cln_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Stfp_ID
- _TimeStudyDt
- _TS_ID

Properties
- Cln_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Stfp_ID
- TimeStudyDt
- TS_ID

Methods
- New

11.6 SA – Clinic Services

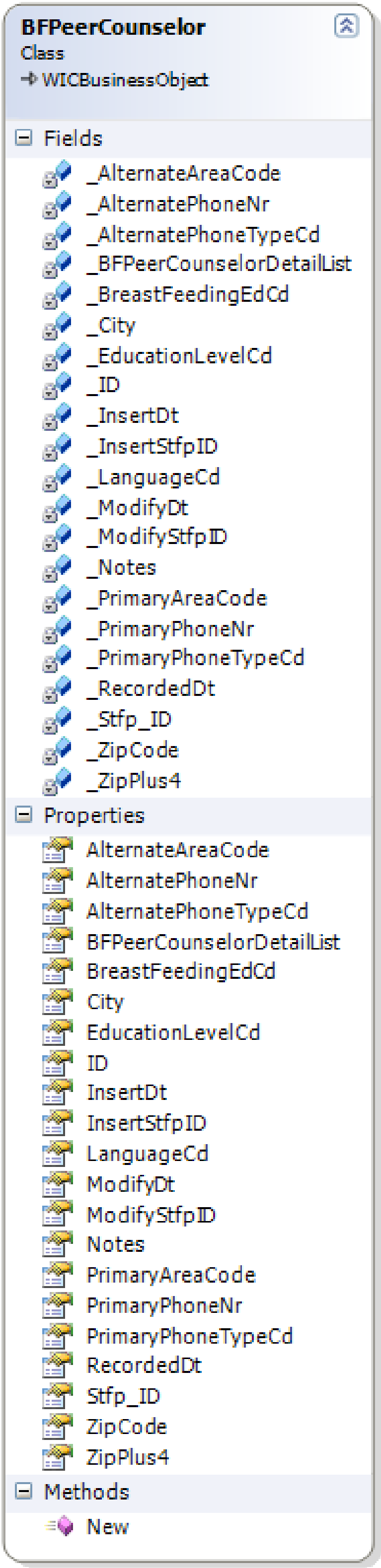


BFPCDocType
Class
➔ WICBusinessObject

Fields
- _DisplaySeqNr
- _DocTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _Status

Properties
- DisplaySeqNr
- DocTypeCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- Status

Methods
- New

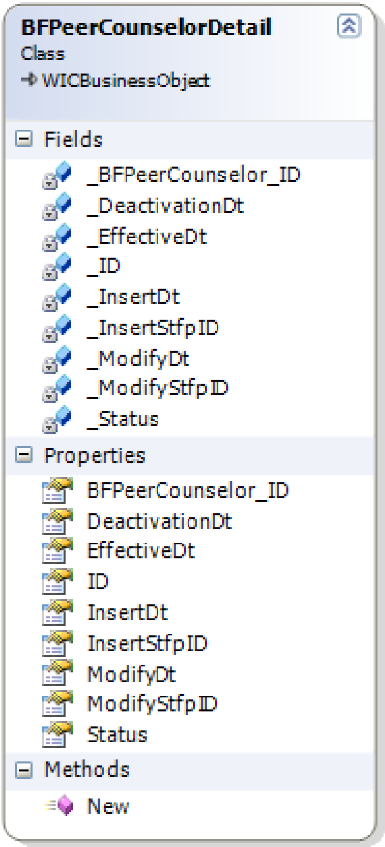


BFPeerCounselor
Class
➔ WICBusinessObject

Fields
- _AlternateAreaCode
- _AlternatePhoneNr
- _AlternatePhoneTypeCd
- _BFPeerCounselorDetailList
- _BreastFeedingEdCd
- _City
- _EducationLevelCd
- _ID
- _InsertDt
- _InsertStfpID
- _LanguageCd
- _ModifyDt
- _ModifyStfpID
- _Notes
- _PrimaryAreaCode
- _PrimaryPhoneNr
- _PrimaryPhoneTypeCd
- _RecordedDt
- _Stfp_ID
- _ZipCode
- _ZipPlus4

Properties
- AlternateAreaCode
- AlternatePhoneNr
- AlternatePhoneTypeCd
- BFPeerCounselorDetailList
- BreastFeedingEdCd
- City
- EducationLevelCd
- ID
- InsertDt
- InsertStfpID
- LanguageCd
- ModifyDt
- ModifyStfpID
- Notes
- PrimaryAreaCode
- PrimaryPhoneNr
- PrimaryPhoneTypeCd
- RecordedDt
- Stfp_ID
- ZipCode
- ZipPlus4

Methods
- New

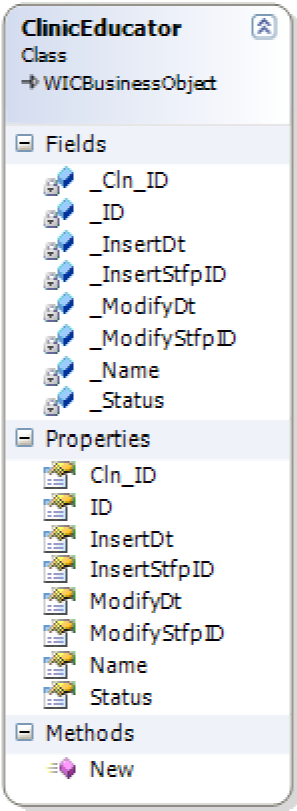


BFPeerCounselorDetail
Class
➔ WICBusinessObject

Fields
- _BFPeerCounselor_ID
- _DeactivationDt
- _EffectiveDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Status

Properties
- BFPeerCounselor_ID
- DeactivationDt
- EffectiveDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Status

Methods
- New

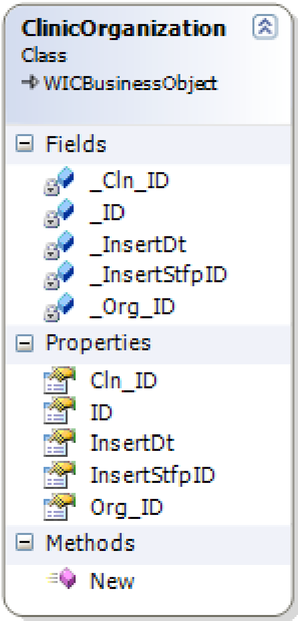


ClinicEducator
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _Status

Properties
- Cln_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- Status

Methods
- New

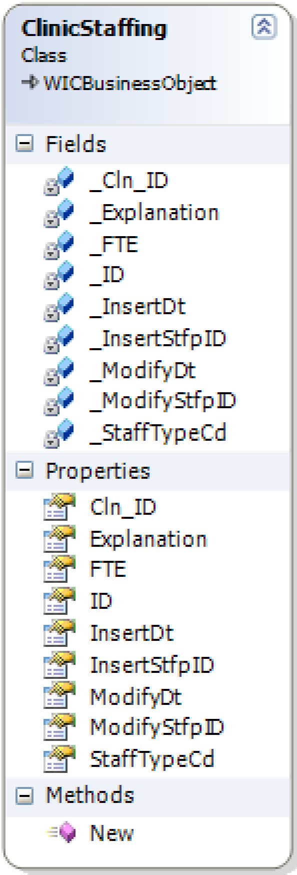


ClinicOrganization
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _ID
- _InsertDt
- _InsertStfpID
- _Org_ID

Properties
- Cln_ID
- ID
- InsertDt
- InsertStfpID
- Org_ID

Methods
- New

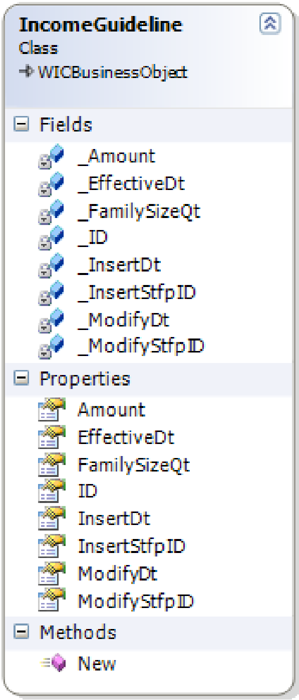


ClinicStaffing
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _Explanation
- _FTE
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _StaffTypeCd

Properties
- Cln_ID
- Explanation
- FTE
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- StaffTypeCd

Methods
- New

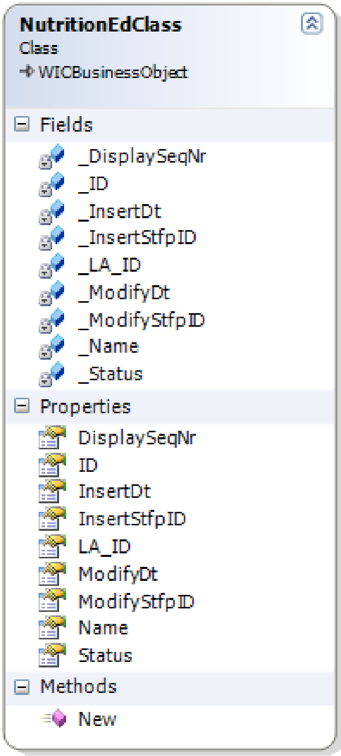


IncomeGuideline
Class
→ WICBusinessObject

Fields
- _Amount
- _EffectiveDt
- _FamilySizeQt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID

Properties
- Amount
- EffectiveDt
- FamilySizeQt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID

Methods
- New

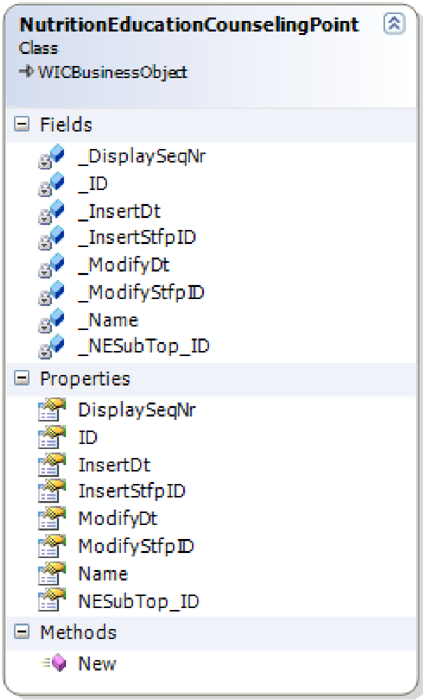


NutritionEdClass
Class
→ WICBusinessObject

Fields
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _Name
- _Status

Properties
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- Name
- Status

Methods
- New

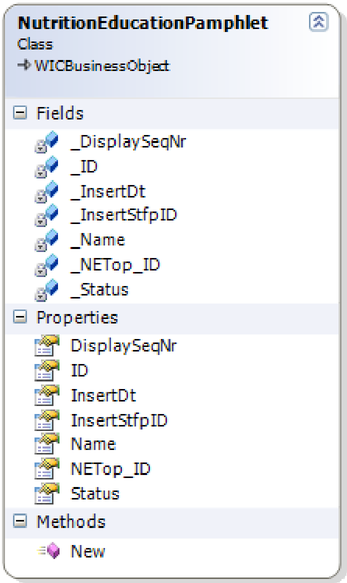


NutritionEducationCounselingPoint
Class
→ WICBusinessObject

Fields
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _NESubTop_ID

Properties
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- NESubTop_ID

Methods
- New

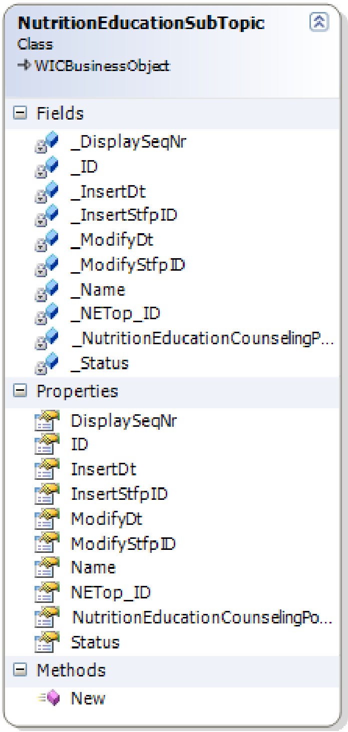


NutritionEducationPamphlet
Class
→ WICBusinessObject

Fields
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _Name
- _NETop_ID
- _Status

Properties
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- Name
- NETop_ID
- Status

Methods
- New

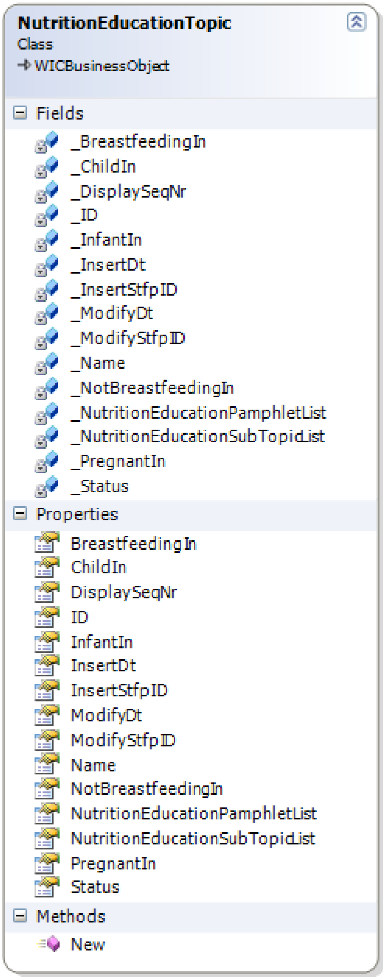


NutritionEducationSubTopic
Class
→ WICBusinessObject

Fields
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _NETop_ID
- _NutritionEducationCounselingP...
- _Status

Properties
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- NETop_ID
- NutritionEducationCounselingPo...
- Status

Methods
- New

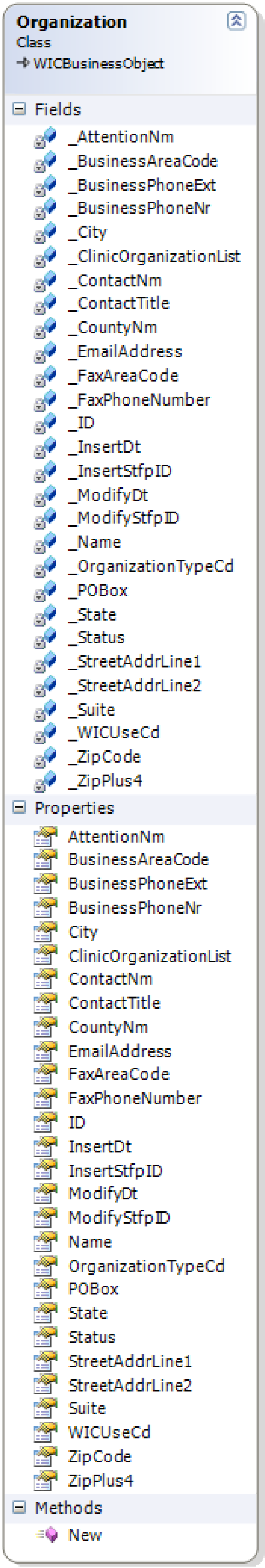


NutritionEducationTopic
Class
→ WICBusinessObject

Fields
- _BreastfeedingIn
- _ChildIn
- _DisplaySeqNr
- _ID
- _InfantIn
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _NotBreastfeedingIn
- _NutritionEducationPamphletList
- _NutritionEducationSubTopicList
- _PregnantIn
- _Status

Properties
- BreastfeedingIn
- ChildIn
- DisplaySeqNr
- ID
- InfantIn
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- NotBreastfeedingIn
- NutritionEducationPamphletList
- NutritionEducationSubTopicList
- PregnantIn
- Status

Methods
- New



Organization
Class
→ WICBusinessObject

⊟ Fields

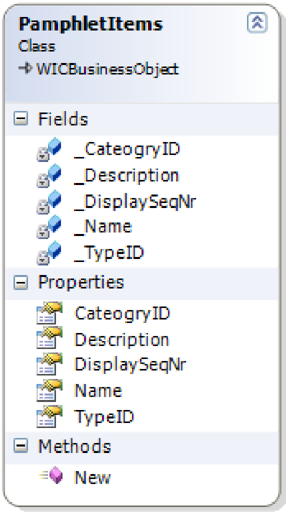
- _AttentionNm
- _BusinessAreaCode
- _BusinessPhoneExt
- _BusinessPhoneNr
- _City
- _ClinicOrganizationList
- _ContactNm
- _ContactTitle
- _CountyNm
- _EmailAddress
- _FaxAreaCode
- _FaxPhoneNumber
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _OrganizationTypeCd
- _POBox
- _State
- _Status
- _StreetAddrLine1
- _StreetAddrLine2
- _Suite
- _WICUseCd
- _ZipCode
- _ZipPlus4

⊟ Properties

- AttentionNm
- BusinessAreaCode
- BusinessPhoneExt
- BusinessPhoneNr
- City
- ClinicOrganizationList
- ContactNm
- ContactTitle
- CountyNm
- EmailAddress
- FaxAreaCode
- FaxPhoneNumber
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- OrganizationTypeCd
- POBox
- State
- Status
- StreetAddrLine1
- StreetAddrLine2
- Suite
- WICUseCd
- ZipCode
- ZipPlus4

⊟ Methods

- New



PamphletItems
Class
→ WICBusinessObject

⊟ Fields

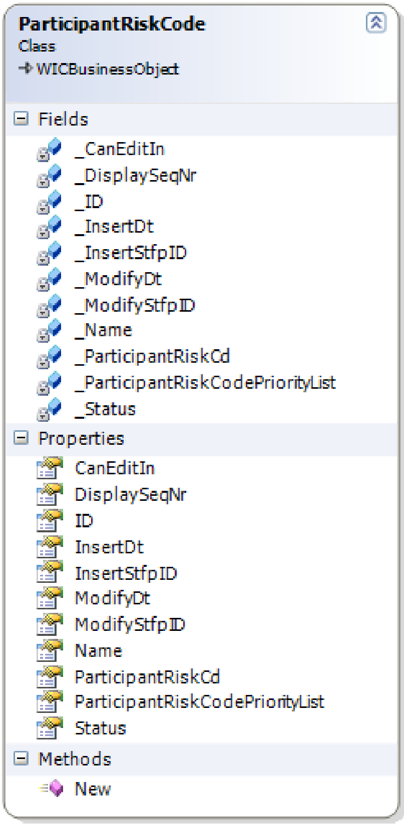
- _CateogryID
- _Description
- _DisplaySeqNr
- _Name
- _TypeID

⊟ Properties

- CateogryID
- Description
- DisplaySeqNr
- Name
- TypeID

⊟ Methods

- New



ParticipantRiskCode
Class
→ WICBusinessObject

⊟ Fields

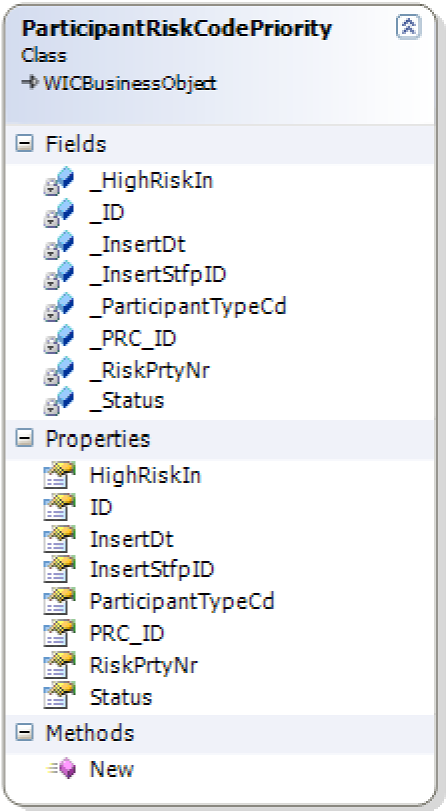
- _CanEditIn
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _ParticipantRiskCd
- _ParticipantRiskCodePriorityList
- _Status

⊟ Properties

- CanEditIn
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- ParticipantRiskCd
- ParticipantRiskCodePriorityList
- Status

⊟ Methods

- New

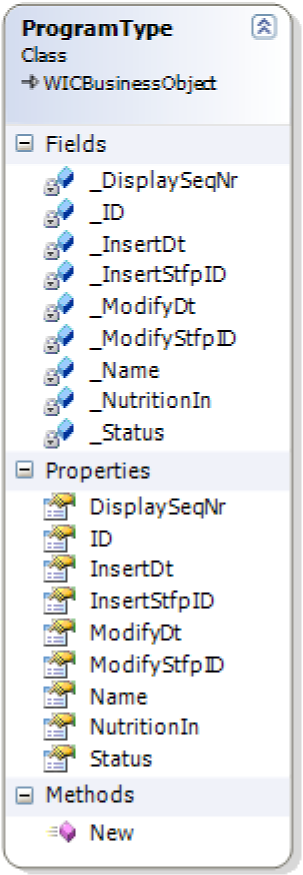


ParticipantRiskCodePriority
Class
→ WICBusinessObject

Fields
- _HighRiskIn
- _ID
- _InsertDt
- _InsertStfpID
- _ParticipantTypeCd
- _PRC_ID
- _RiskPrtyNr
- _Status

Properties
- HighRiskIn
- ID
- InsertDt
- InsertStfpID
- ParticipantTypeCd
- PRC_ID
- RiskPrtyNr
- Status

Methods
- New

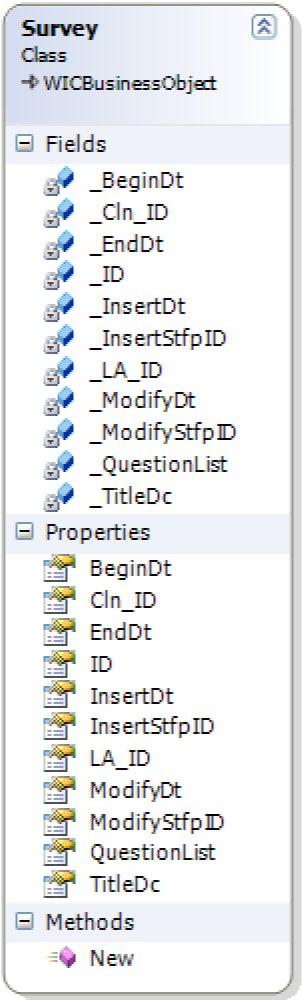


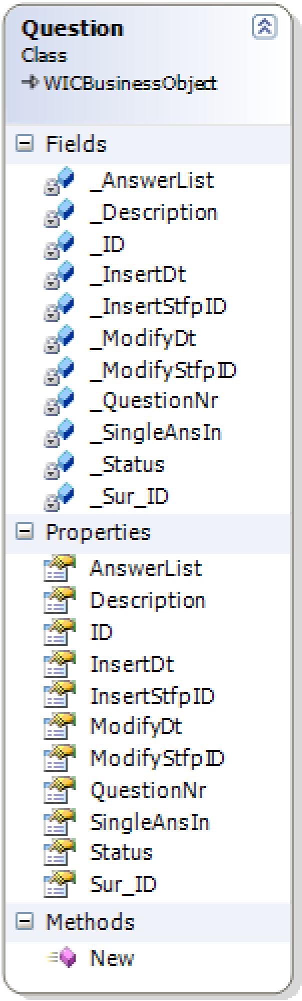
ProgramType
Class
→ WICBusinessObject

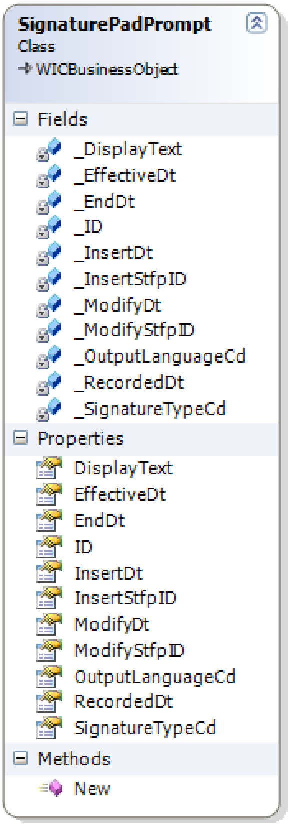
Fields
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _NutritionIn
- _Status

Properties
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- NutritionIn
- Status

Methods
- New


Survey
Class
WICBusinessObject
Fields
_BeginDt
_Cln_ID
_EndDt
_ID
_InsertDt
_InsertStfpID
_LA_ID
_ModifyDt
_ModifyStfpID
_QuestionList
_TitleDc
Properties
BeginDt
Cln_ID
EndDt
ID
InsertDt
InsertStfpID
LA_ID
ModifyDt
ModifyStfpID
QuestionList
TitleDc
Methods
New


Question
Class
WICBusinessObject
Fields
_AnswerList
_Description
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_QuestionNr
_SingleAnsIn
_Status
_Sur_ID
Properties
AnswerList
Description
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
QuestionNr
SingleAnsIn
Status
Sur_ID
Methods
New

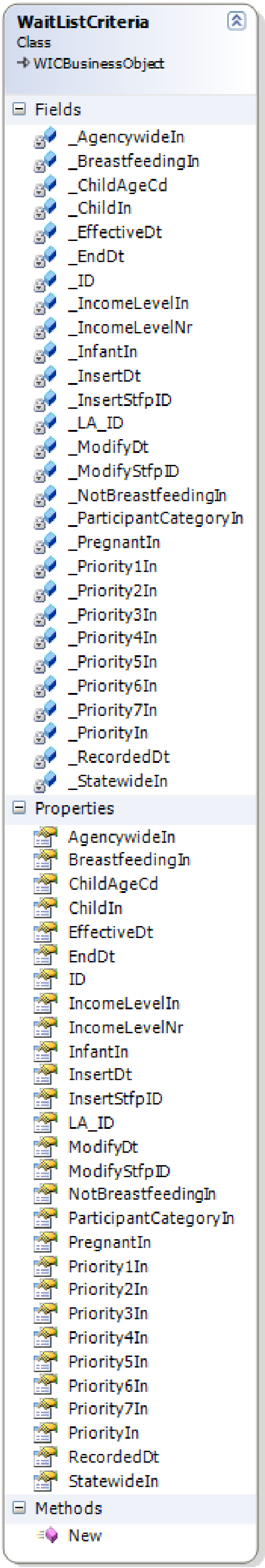


SignaturePadPrompt
Class
→ WICBusinessObject

Fields
- _DisplayText
- _EffectiveDt
- _EndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OutputLanguageCd
- _RecordedDt
- _SignatureTypeCd

Properties
- DisplayText
- EffectiveDt
- EndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OutputLanguageCd
- RecordedDt
- SignatureTypeCd

Methods
- New



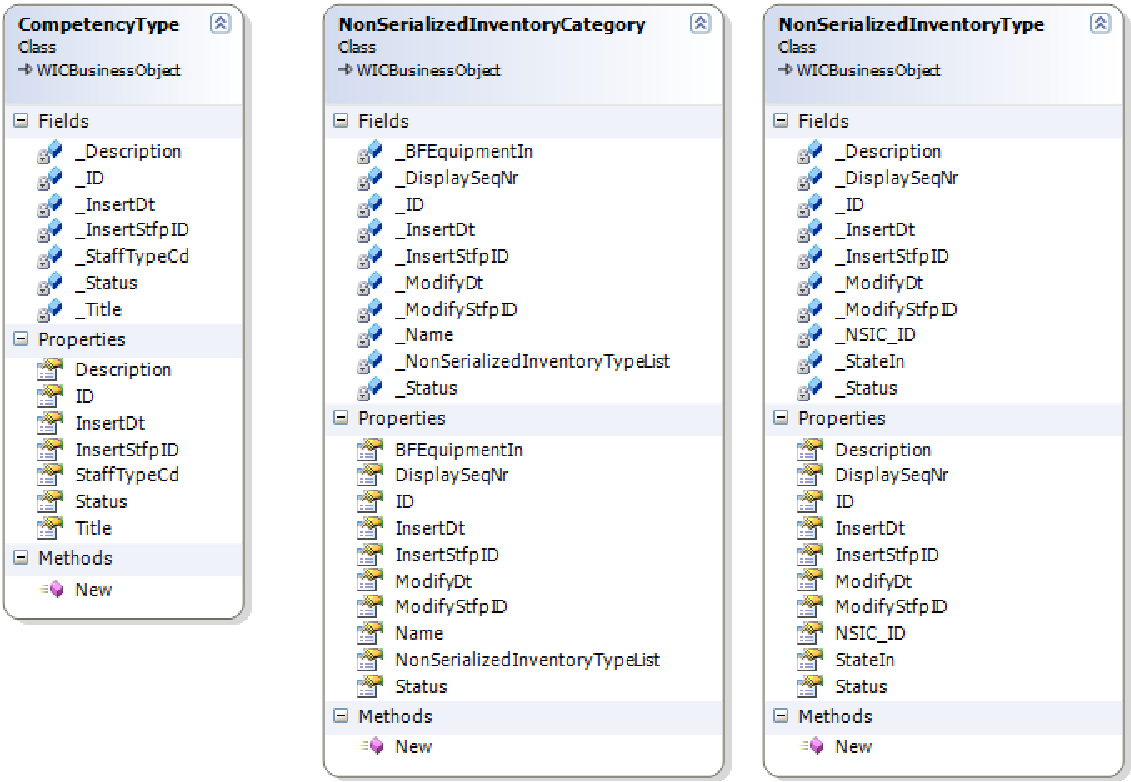
WaitListCriteria
Class
→ WICBusinessObject

Fields
- _AgencywideIn
- _BreastfeedingIn
- _ChildAgeCd
- _ChildIn
- _EffectiveDt
- _EndDt
- _ID
- _IncomeLevelIn
- _IncomeLevelNr
- _InfantIn
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _NotBreastfeedingIn
- _ParticipantCategoryIn
- _PregnantIn
- _Priority1In
- _Priority2In
- _Priority3In
- _Priority4In
- _Priority5In
- _Priority6In
- _Priority7In
- _PriorityIn
- _RecordedDt
- _StatewideIn

Properties
- AgencywideIn
- BreastfeedingIn
- ChildAgeCd
- ChildIn
- EffectiveDt
- EndDt
- ID
- IncomeLevelIn
- IncomeLevelNr
- InfantIn
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- NotBreastfeedingIn
- ParticipantCategoryIn
- PregnantIn
- Priority1In
- Priority2In
- Priority3In
- Priority4In
- Priority5In
- Priority6In
- Priority7In
- PriorityIn
- RecordedDt
- StatewideIn

Methods
- New

11.7 SA – Operations



CompetencyType
Class
→ WICBusinessObject

Fields
- _Description
- _ID
- _InsertDt
- _InsertStfpID
- _StaffTypeCd
- _Status
- _Title

Properties
- Description
- ID
- InsertDt
- InsertStfpID
- StaffTypeCd
- Status
- Title

Methods
- New

NonSerializedInventoryCategory
Class
→ WICBusinessObject

Fields
- _BFEquipmentIn
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _NonSerializedInventoryTypeList
- _Status

Properties
- BFEquipmentIn
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- NonSerializedInventoryTypeList
- Status

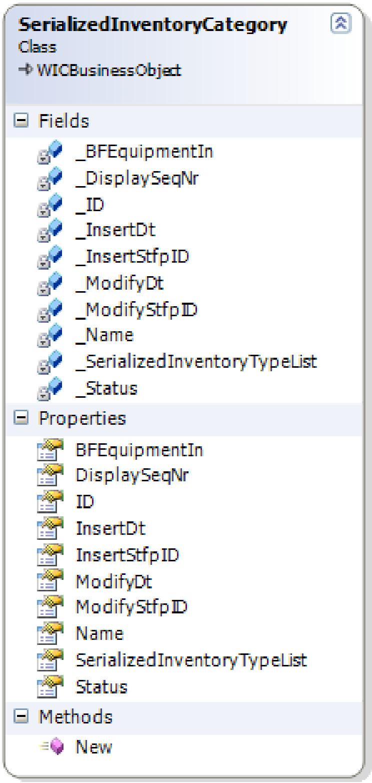
Methods
- New

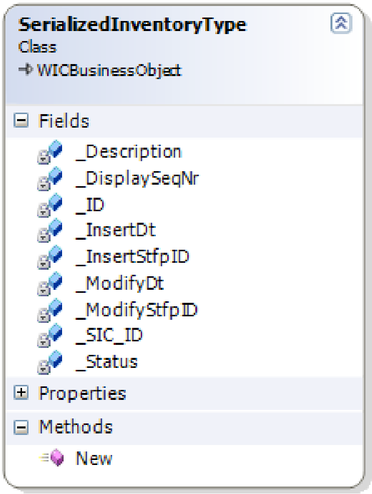
NonSerializedInventoryType
Class
→ WICBusinessObject

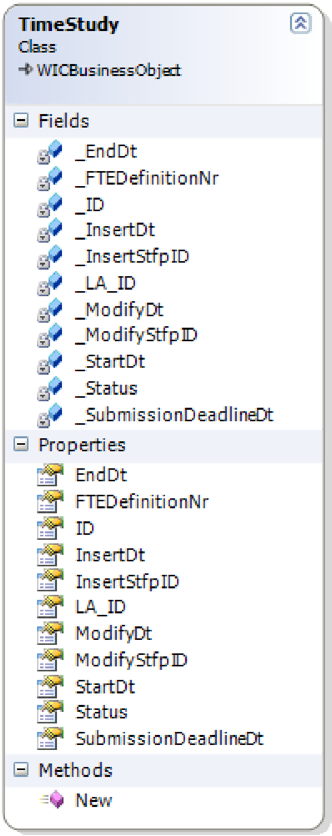
Fields
- _Description
- _DisplaySeqNr
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NSIC_ID
- _StateIn
- _Status

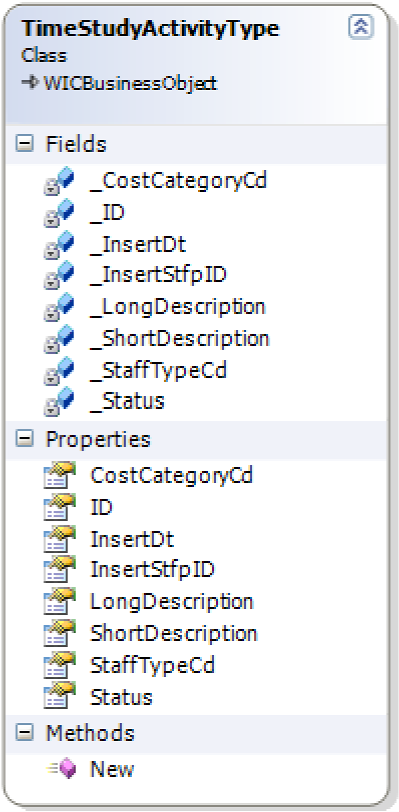
Properties
- Description
- DisplaySeqNr
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NSIC_ID
- StateIn
- Status

Methods
- New


SerializedInventoryCategory
Class
WICBusinessObject
Fields
_BFEquipmentIn
_DisplaySeqNr
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_Name
_SerializedInventoryTypeList
_Status
Properties
BFEquipmentIn
DisplaySeqNr
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Name
SerializedInventoryTypeList
Status
Methods
New


SerializedInventoryType
Class
WICBusinessObject
Fields
_Description
_DisplaySeqNr
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_SIC_ID
_Status
Properties
Methods
New


TimeStudy
Class
WICBusinessObject
Fields
_EndDt
_FTEDefinitionNr
_ID
_InsertDt
_InsertStfpID
_LA_ID
_ModifyDt
_ModifyStfpID
_StartDt
_Status
_SubmissionDeadlineDt
Properties
EndDt
FTEDefinitionNr
ID
InsertDt
InsertStfpID
LA_ID
ModifyDt
ModifyStfpID
StartDt
Status
SubmissionDeadlineDt
Methods
New



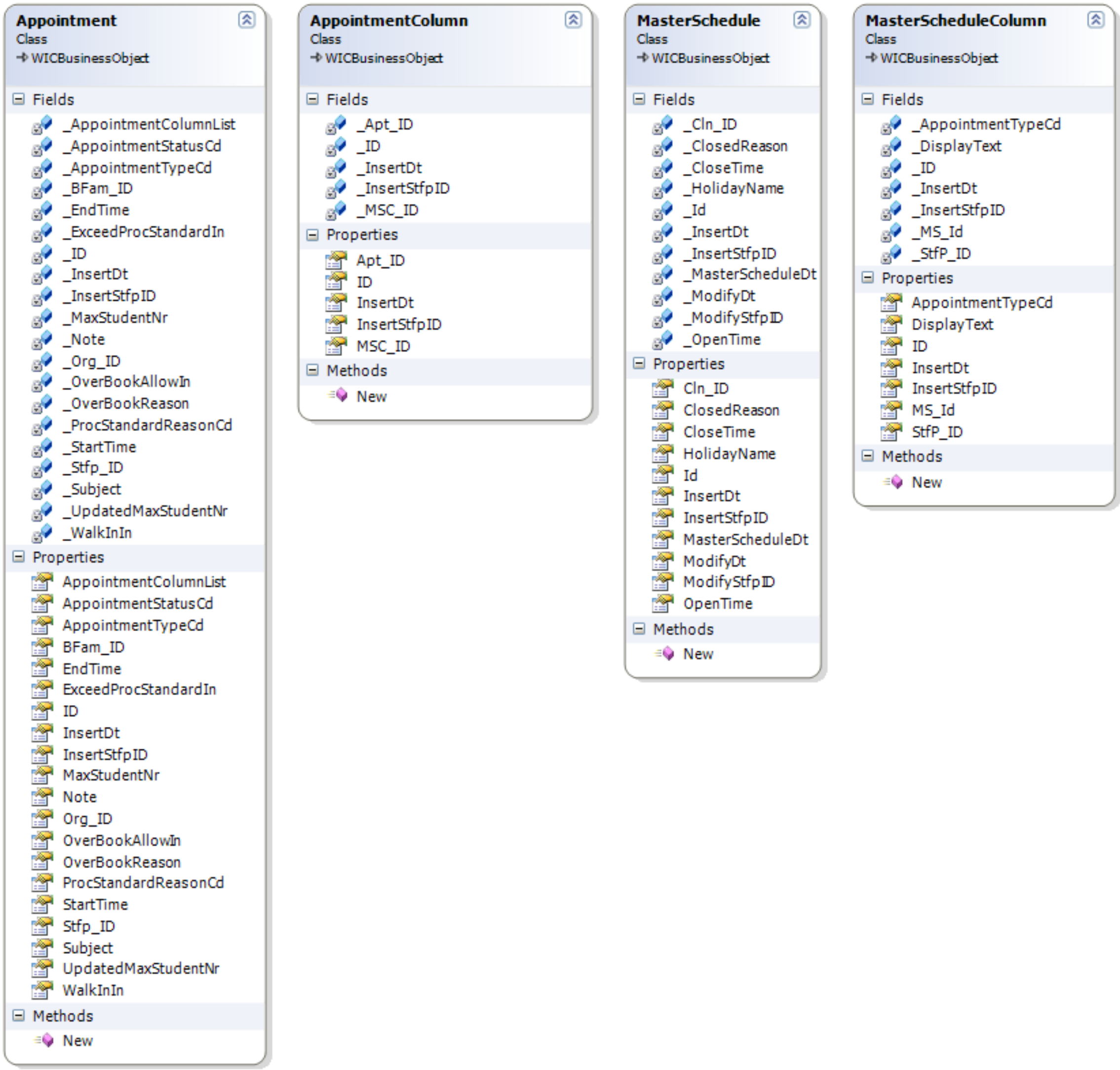
TimeStudyActivityType
Class
→ WICBusinessObject

Fields
- _CostCategoryCd
- _ID
- _InsertDt
- _InsertStfpID
- _LongDescription
- _ShortDescription
- _StaffTypeCd
- _Status

Properties
- CostCategoryCd
- ID
- InsertDt
- InsertStfpID
- LongDescription
- ShortDescription
- StaffTypeCd
- Status

Methods
- New

11.8 SA – Scheduler



Appointment
Class
→ WICBusinessObject

Fields
- _AppointmentColumnList
- _AppointmentStatusCd
- _AppointmentTypeCd
- _BFam_ID
- _EndTime
- _ExceedProcStandardIn
- _ID
- _InsertDt
- _InsertStfpID
- _MaxStudentNr
- _Note
- _Org_ID
- _OverBookAllowIn
- _OverBookReason
- _ProcStandardReasonCd
- _StartTime
- _Stfp_ID
- _Subject
- _UpdatedMaxStudentNr
- _WalkInIn

Properties
- AppointmentColumnList
- AppointmentStatusCd
- AppointmentTypeCd
- BFam_ID
- EndTime
- ExceedProcStandardIn
- ID
- InsertDt
- InsertStfpID
- MaxStudentNr
- Note
- Org_ID
- OverBookAllowIn
- OverBookReason
- ProcStandardReasonCd
- StartTime
- Stfp_ID
- Subject
- UpdatedMaxStudentNr
- WalkInIn

Methods
- New

AppointmentColumn
Class
→ WICBusinessObject

Fields
- _Apt_ID
- _ID
- _InsertDt
- _InsertStfpID
- _MSC_ID

Properties
- Apt_ID
- ID
- InsertDt
- InsertStfpID
- MSC_ID

Methods
- New

MasterSchedule
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _ClosedReason
- _CloseTime
- _HolidayName
- _Id
- _InsertDt
- _InsertStfpID
- _MasterScheduleDt
- _ModifyDt
- _ModifyStfpID
- _OpenTime

Properties
- Cln_ID
- ClosedReason
- CloseTime
- HolidayName
- Id
- InsertDt
- InsertStfpID
- MasterScheduleDt
- ModifyDt
- ModifyStfpID
- OpenTime

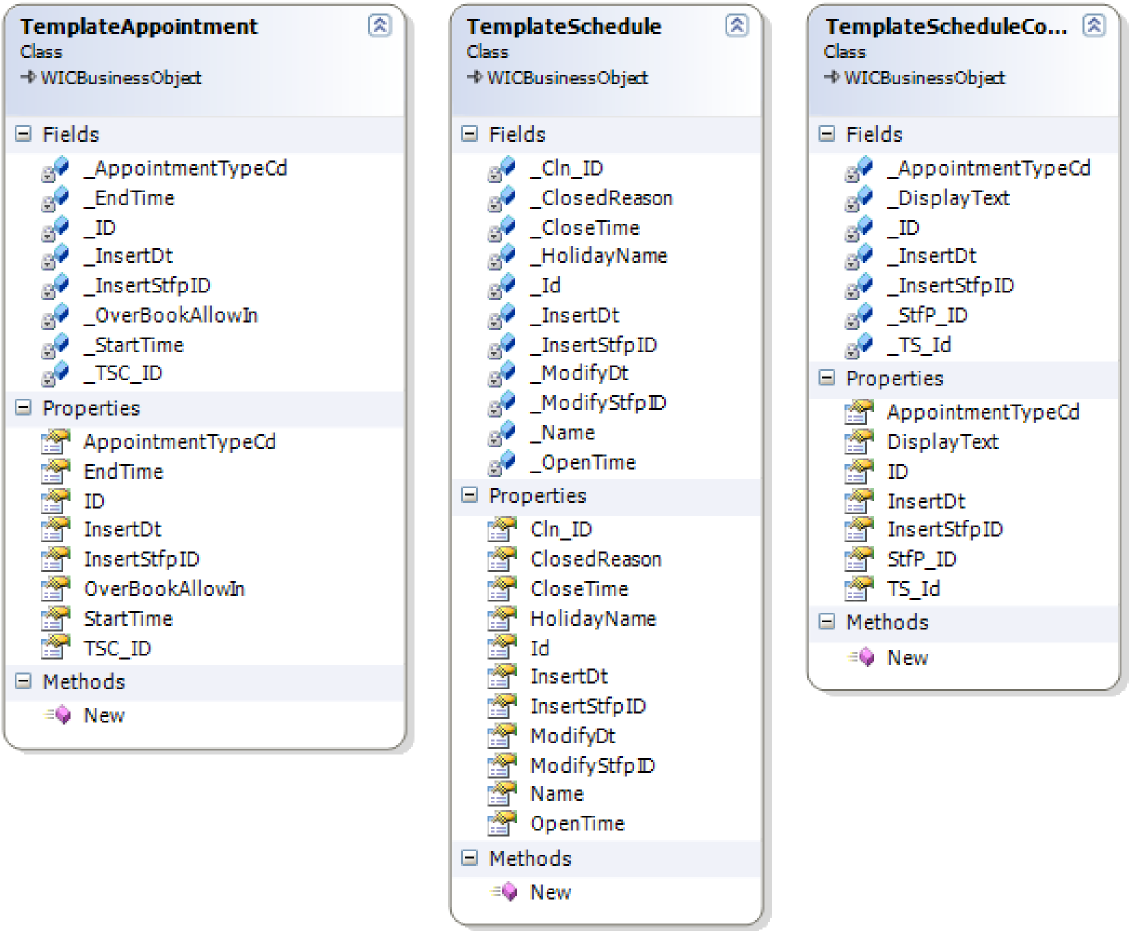
Methods
- New

MasterScheduleColumn
Class
→ WICBusinessObject

Fields
- _AppointmentTypeCd
- _DisplayText
- _ID
- _InsertDt
- _InsertStfpID
- _MS_Id
- _StfP_ID

Properties
- AppointmentTypeCd
- DisplayText
- ID
- InsertDt
- InsertStfpID
- MS_Id
- StfP_ID

Methods
- New



TemplateAppointment
Class
→ WICBusinessObject

Fields
- _AppointmentTypeCd
- _EndTime
- _ID
- _InsertDt
- _InsertStfpID
- _OverBookAllowIn
- _StartTime
- _TSC_ID

Properties
- AppointmentTypeCd
- EndTime
- ID
- InsertDt
- InsertStfpID
- OverBookAllowIn
- StartTime
- TSC_ID

Methods
- New

TemplateSchedule
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _ClosedReason
- _CloseTime
- _HolidayName
- _Id
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _OpenTime

Properties
- Cln_ID
- ClosedReason
- CloseTime
- HolidayName
- Id
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- OpenTime

Methods
- New

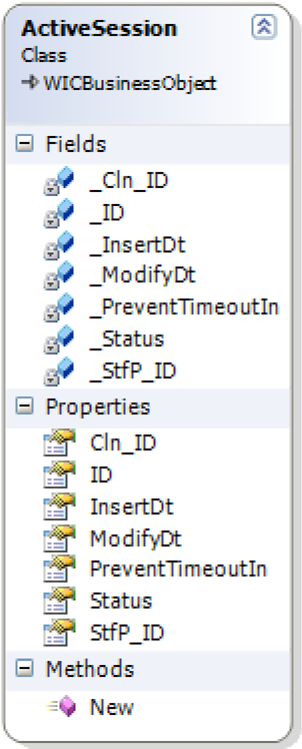
TemplateScheduleCo...
Class
→ WICBusinessObject

Fields
- _AppointmentTypeCd
- _DisplayText
- _ID
- _InsertDt
- _InsertStfpID
- _StfP_ID
- _TS_Id

Properties
- AppointmentTypeCd
- DisplayText
- ID
- InsertDt
- InsertStfpID
- StfP_ID
- TS_Id

Methods
- New

11.9 SA – System Wide

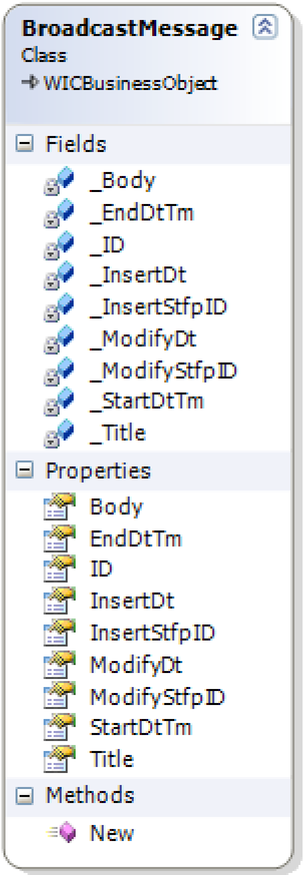


ActiveSession
Class
→ WICBusinessObject

Fields
_Cln_ID
_ID
_InsertDt
_ModifyDt
_PreventTimeoutIn
_Status
_StfP_ID

Properties
Cln_ID
ID
InsertDt
ModifyDt
PreventTimeoutIn
Status
StfP_ID

Methods
New

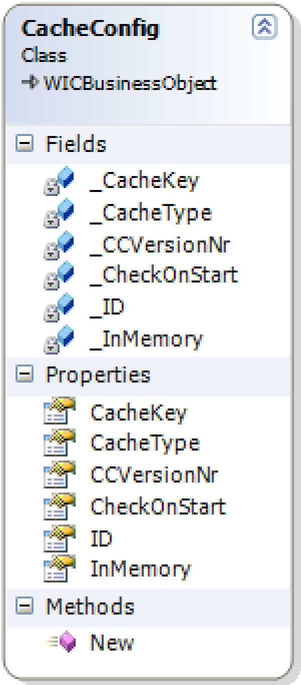


BroadcastMessage
Class
→ WICBusinessObject

Fields
_Body
_EndDtTm
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_StartDtTm
_Title

Properties
Body
EndDtTm
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
StartDtTm
Title

Methods
New

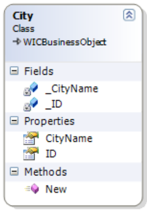


CacheConfig
Class
→ WICBusinessObject

Fields
_CacheKey
_CacheType
_CCVersionNr
_CheckOnStart
_ID
_InMemory

Properties
CacheKey
CacheType
CCVersionNr
CheckOnStart
ID
InMemory

Methods
New

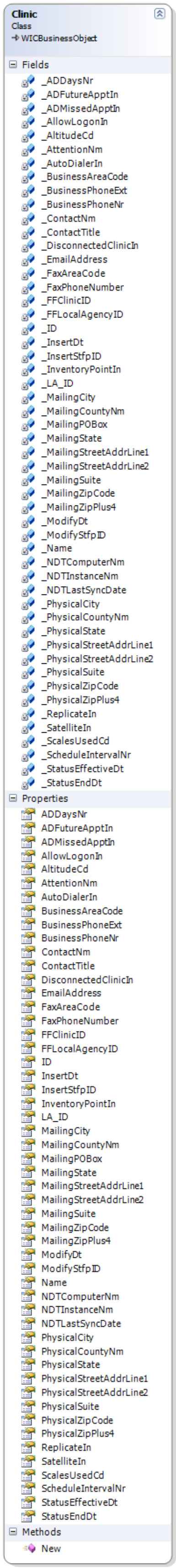


City
Class
→ WICBusinessObject

Fields
_CityName
_ID

Properties
CityName
ID

Methods
New

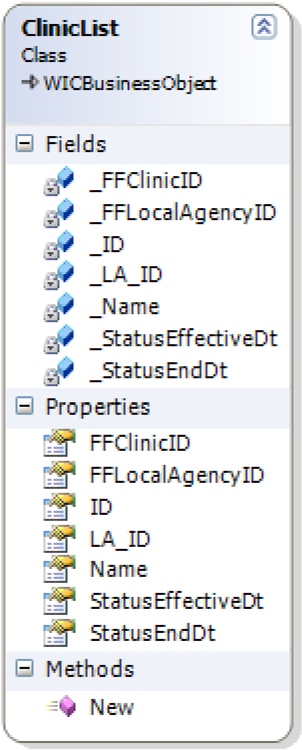


Clinic
Class
→ WICBusinessObject

Fields
_ADDaysNr
_ADFutureApptIn
_ADMissedApptIn
_AllowLogonIn
_AltitudeCd
_AttentionNm
_AutoDialerIn
_BusinessAreaCode
_BusinessPhoneExt
_BusinessPhoneNr
_ContactNm
_ContactTitle
_DisconnectedClinicIn
_EmailAddress
_FaxAreaCode
_FaxPhoneNumber
_FFClinicID
_FFLocalAgencyID
_ID
_InsertDt
_InsertStfpID
_InventoryPointIn
_LA_ID
_MailingCity
_MailingCountyNm
_MailingPOBox
_MailingState
_MailingStreetAddrLine1
_MailingStreetAddrLine2
_MailingSuite
_MailingZipCode
_MailingZipPlus4
_ModifyDt
_ModifyStfpID
_Name
_NDTComputerNm
_NDTInstanceNm
_NDTLastSyncDate
_PhysicalCity
_PhysicalCountyNm
_PhysicalState
_PhysicalStreetAddrLine1
_PhysicalStreetAddrLine2
_PhysicalSuite
_PhysicalZipCode
_PhysicalZipPlus4
_ReplicateIn
_SatelliteIn
_ScalesUsedCd
_ScheduleIntervalNr
_StatusEffectiveDt
_StatusEndDt

Properties
ADDaysNr
ADFutureApptIn
ADMissedApptIn
AllowLogonIn
AltitudeCd
AttentionNm
AutoDialerIn
BusinessAreaCode
BusinessPhoneExt
BusinessPhoneNr
ContactNm
ContactTitle
DisconnectedClinicIn
EmailAddress
FaxAreaCode
FaxPhoneNumber
FFClinicID
FFLocalAgencyID
ID
InsertDt
InsertStfpID
InventoryPointIn
LA_ID
MailingCity
MailingCountyNm
MailingPOBox
MailingState
MailingStreetAddrLine1
MailingStreetAddrLine2
MailingSuite
MailingZipCode
MailingZipPlus4
ModifyDt
ModifyStfpID
Name
NDTComputerNm
NDTInstanceNm
NDTLastSyncDate
PhysicalCity
PhysicalCountyNm
PhysicalState
PhysicalStreetAddrLine1
PhysicalStreetAddrLine2
PhysicalSuite
PhysicalZipCode
PhysicalZipPlus4
ReplicateIn
SatelliteIn
ScalesUsedCd
ScheduleIntervalNr
StatusEffectiveDt
StatusEndDt

Methods
New

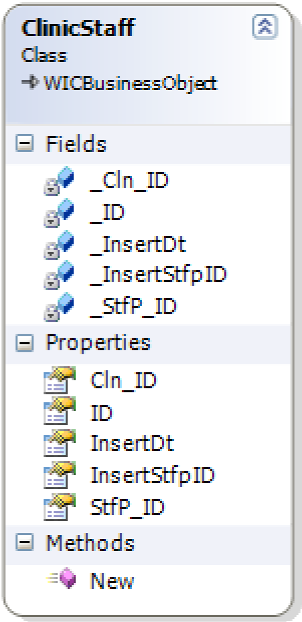


ClinicList
Class
→ WICBusinessObject

⊟ Fields
- _FFClinicID
- _FFLocalAgencyID
- _ID
- _LA_ID
- _Name
- _StatusEffectiveDt
- _StatusEndDt

⊟ Properties
- FFClinicID
- FFLocalAgencyID
- ID
- LA_ID
- Name
- StatusEffectiveDt
- StatusEndDt

⊟ Methods
- New

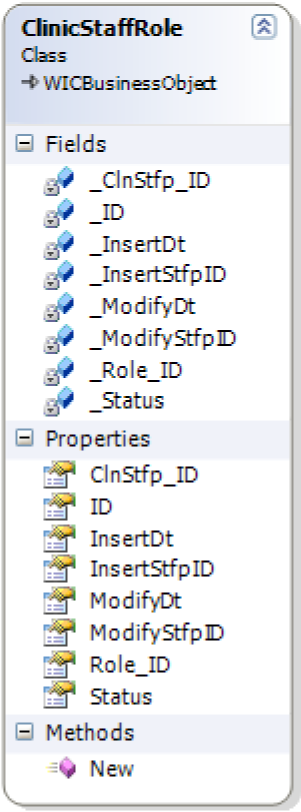


ClinicStaff
Class
→ WICBusinessObject

⊟ Fields
- _Cln_ID
- _ID
- _InsertDt
- _InsertStfpID
- _StfP_ID

⊟ Properties
- Cln_ID
- ID
- InsertDt
- InsertStfpID
- StfP_ID

⊟ Methods
- New

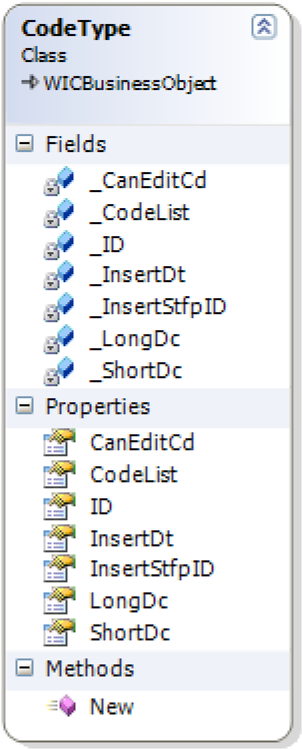


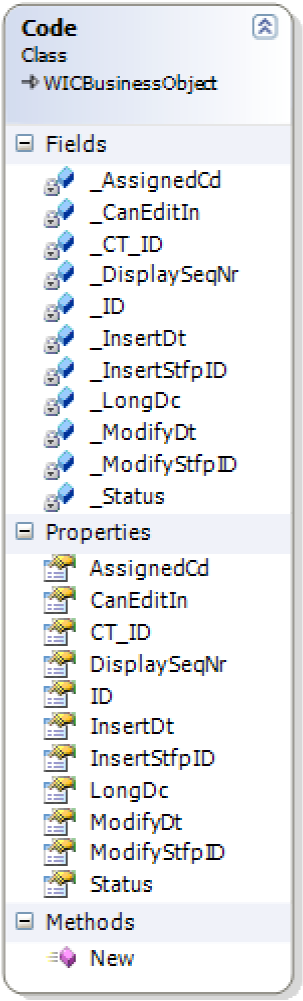
ClinicStaffRole
Class
→ WICBusinessObject

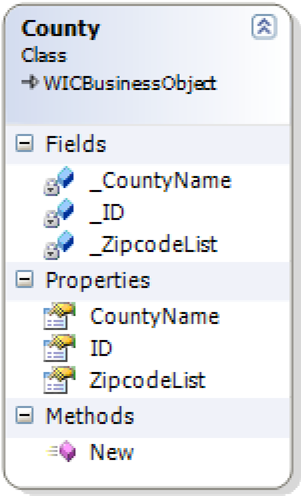
⊟ Fields
- _ClnStfp_ID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Role_ID
- _Status

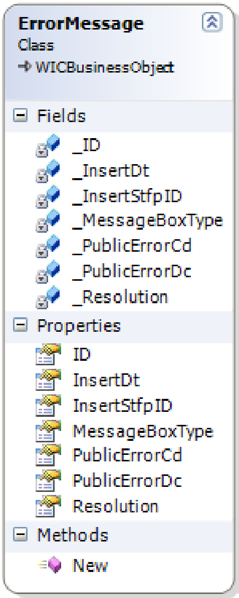
⊟ Properties
- ClnStfp_ID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Role_ID
- Status

⊟ Methods
- New


CodeType
Class
WICBusinessObject
Fields
_CanEditCd
_CodeList
_ID
_InsertDt
_InsertStfpID
_LongDc
_ShortDc
Properties
CanEditCd
CodeList
ID
InsertDt
InsertStfpID
LongDc
ShortDc
Methods
New


Code
Class
WICBusinessObject
Fields
_AssignedCd
_CanEditIn
_CT_ID
_DisplaySeqNr
_ID
_InsertDt
_InsertStfpID
_LongDc
_ModifyDt
_ModifyStfpID
_Status
Properties
AssignedCd
CanEditIn
CT_ID
DisplaySeqNr
ID
InsertDt
InsertStfpID
LongDc
ModifyDt
ModifyStfpID
Status
Methods
New


County
Class
WICBusinessObject
Fields
_CountyName
_ID
_ZipcodeList
Properties
CountyName
ID
ZipcodeList
Methods
New

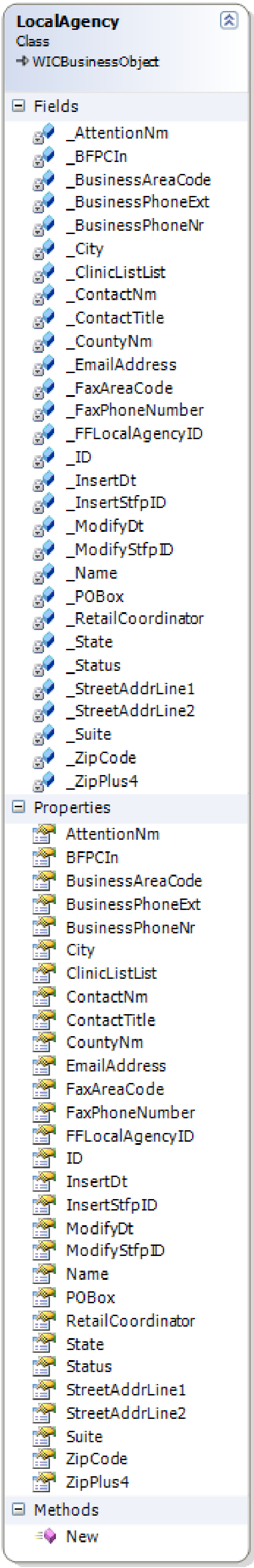


ErrorMessage
Class
→ WICBusinessObject

Fields
- _ID
- _InsertDt
- _InsertStfpID
- _MessageBoxType
- _PublicErrorCd
- _PublicErrorDc
- _Resolution

Properties
- ID
- InsertDt
- InsertStfpID
- MessageBoxType
- PublicErrorCd
- PublicErrorDc
- Resolution

Methods
- New

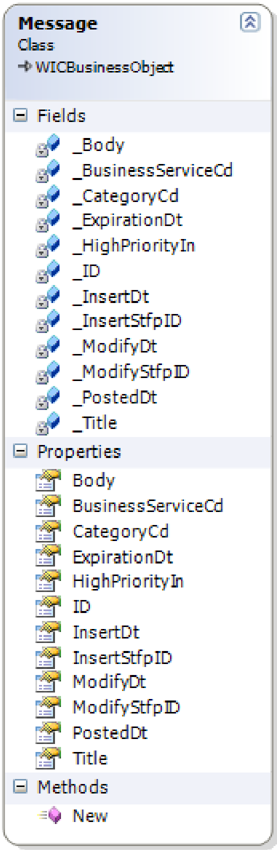


LocalAgency
Class
→ WICBusinessObject

Fields
- _AttentionNm
- _BFPCIn
- _BusinessAreaCode
- _BusinessPhoneExt
- _BusinessPhoneNr
- _City
- _ClinicListList
- _ContactNm
- _ContactTitle
- _CountyNm
- _EmailAddress
- _FaxAreaCode
- _FaxPhoneNumber
- _FFLocalAgencyID
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _POBox
- _RetailCoordinator
- _State
- _Status
- _StreetAddrLine1
- _StreetAddrLine2
- _Suite
- _ZipCode
- _ZipPlus4

Properties
- AttentionNm
- BFPCIn
- BusinessAreaCode
- BusinessPhoneExt
- BusinessPhoneNr
- City
- ClinicListList
- ContactNm
- ContactTitle
- CountyNm
- EmailAddress
- FaxAreaCode
- FaxPhoneNumber
- FFLocalAgencyID
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- POBox
- RetailCoordinator
- State
- Status
- StreetAddrLine1
- StreetAddrLine2
- Suite
- ZipCode
- ZipPlus4

Methods
- New

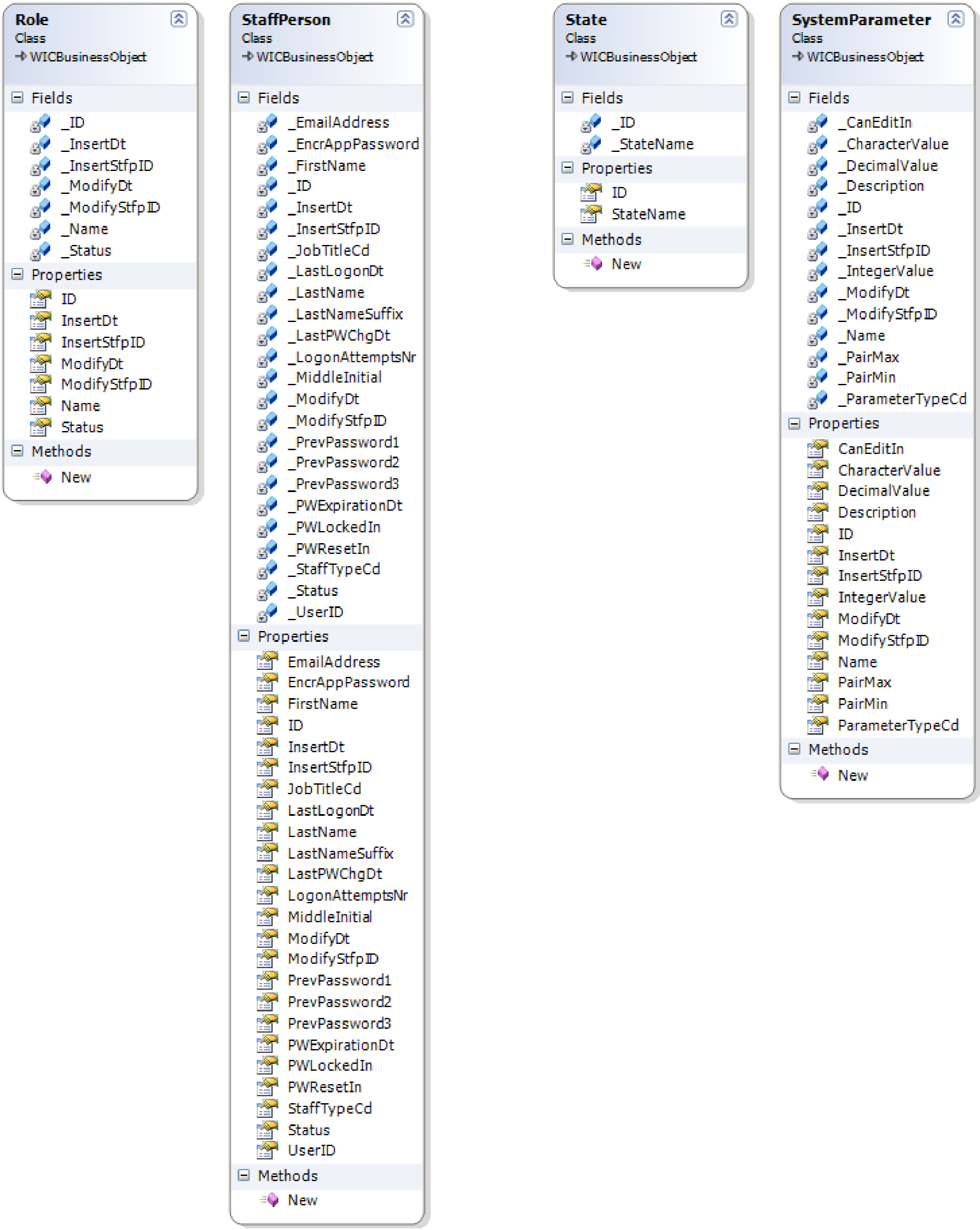


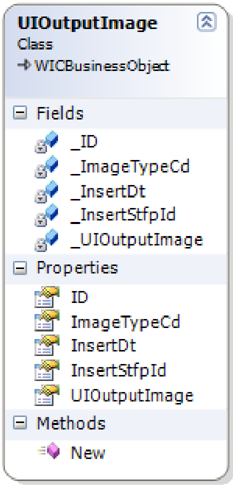
Message
Class
→ WICBusinessObject

Fields
- _Body
- _BusinessServiceCd
- _CategoryCd
- _ExpirationDt
- _HighPriorityIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _PostedDt
- _Title

Properties
- Body
- BusinessServiceCd
- CategoryCd
- ExpirationDt
- HighPriorityIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- PostedDt
- Title

Methods
- New


Role
Class
→ WICBusinessObject
Fields
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_Name
_Status
Properties
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
Name
Status
Methods
New
StaffPerson
Class
→ WICBusinessObject
Fields
_EmailAddress
_EncrAppPassword
_FirstName
_ID
_InsertDt
_InsertStfpID
_JobTitleCd
_LastLogonDt
_LastName
_LastNameSuffix
_LastPWChgDt
_LogonAttemptsNr
_MiddleInitial
_ModifyDt
_ModifyStfpID
_PrevPassword1
_PrevPassword2
_PrevPassword3
_PWExpirationDt
_PWLockedIn
_PWResetIn
_StaffTypeCd
_Status
_UserID
Properties
EmailAddress
EncrAppPassword
FirstName
ID
InsertDt
InsertStfpID
JobTitleCd
LastLogonDt
LastName
LastNameSuffix
LastPWChgDt
LogonAttemptsNr
MiddleInitial
ModifyDt
ModifyStfpID
PrevPassword1
PrevPassword2
PrevPassword3
PWExpirationDt
PWLockedIn
PWResetIn
StaffTypeCd
Status
UserID
Methods
New
State
Class
→ WICBusinessObject
Fields
_ID
_StateName
Properties
ID
StateName
Methods
New
SystemParameter
Class
→ WICBusinessObject
Fields
_CanEditIn
_CharacterValue
_DecimalValue
_Description
_ID
_InsertDt
_InsertStfpID
_IntegerValue
_ModifyDt
_ModifyStfpID
_Name
_PairMax
_PairMin
_ParameterTypeCd
Properties
CanEditIn
CharacterValue
DecimalValue
Description
ID
InsertDt
InsertStfpID
IntegerValue
ModifyDt
ModifyStfpID
Name
PairMax
PairMin
ParameterTypeCd
Methods
New

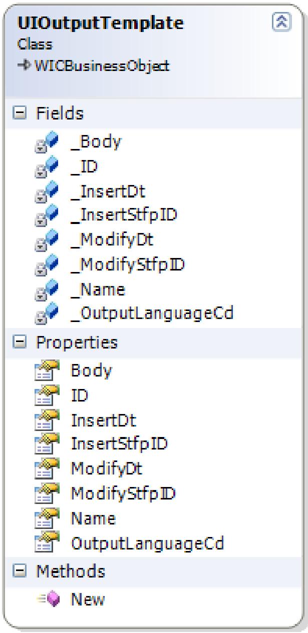


UIOutputImage
Class
→ WICBusinessObject

Fields
- _ID
- _ImageTypeCd
- _InsertDt
- _InsertStfpId
- _UIOutputImage

Properties
- ID
- ImageTypeCd
- InsertDt
- InsertStfpId
- UIOutputImage

Methods
- New

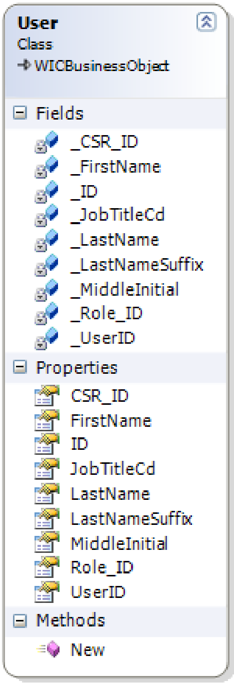


UIOutputTemplate
Class
→ WICBusinessObject

Fields
- _Body
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _OutputLanguageCd

Properties
- Body
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- OutputLanguageCd

Methods
- New

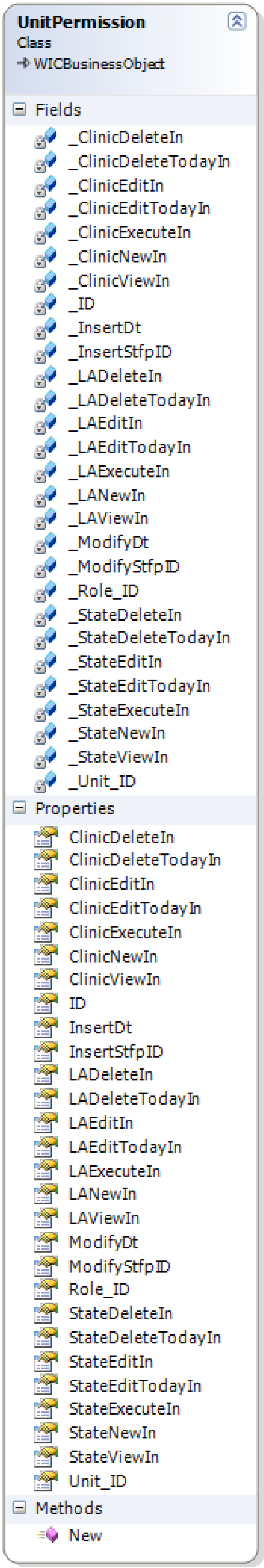


User
Class
→ WICBusinessObject

Fields
- _CSR_ID
- _FirstName
- _ID
- _JobTitleCd
- _LastName
- _LastNameSuffix
- _MiddleInitial
- _Role_ID
- _UserID

Properties
- CSR_ID
- FirstName
- ID
- JobTitleCd
- LastName
- LastNameSuffix
- MiddleInitial
- Role_ID
- UserID

Methods
- New

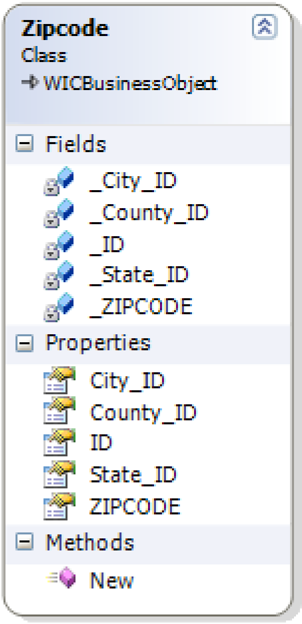


UnitPermission
Class
→ WICBusinessObject

Fields
- _ClinicDeleteIn
- _ClinicDeleteTodayIn
- _ClinicEditIn
- _ClinicEditTodayIn
- _ClinicExecuteIn
- _ClinicNewIn
- _ClinicViewIn
- _ID
- _InsertDt
- _InsertStfpID
- _LADeleteIn
- _LADeleteTodayIn
- _LAEditIn
- _LAEditTodayIn
- _LAExecuteIn
- _LANewIn
- _LAViewIn
- _ModifyDt
- _ModifyStfpID
- _Role_ID
- _StateDeleteIn
- _StateDeleteTodayIn
- _StateEditIn
- _StateEditTodayIn
- _StateExecuteIn
- _StateNewIn
- _StateViewIn
- _Unit_ID

Properties
- ClinicDeleteIn
- ClinicDeleteTodayIn
- ClinicEditIn
- ClinicEditTodayIn
- ClinicExecuteIn
- ClinicNewIn
- ClinicViewIn
- ID
- InsertDt
- InsertStfpID
- LADeleteIn
- LADeleteTodayIn
- LAEditIn
- LAEditTodayIn
- LAExecuteIn
- LANewIn
- LAViewIn
- ModifyDt
- ModifyStfpID
- Role_ID
- StateDeleteIn
- StateDeleteTodayIn
- StateEditIn
- StateEditTodayIn
- StateExecuteIn
- StateNewIn
- StateViewIn
- Unit_ID

Methods
- New



Zipcode
Class
→ WICBusinessObject

⊟ Fields

- _City_ID
- _County_ID
- _ID
- _State_ID
- _ZIPCODE

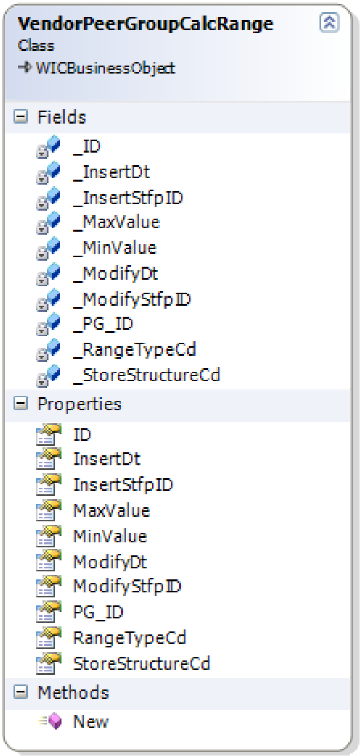
⊟ Properties

- City_ID
- County_ID
- ID
- State_ID
- ZIPCODE

⊟ Methods

- New

11.10SA – Vendor

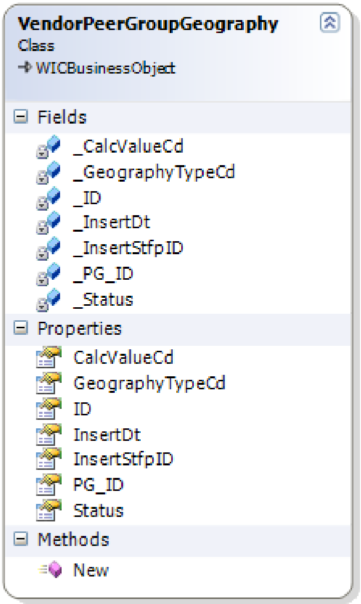


VendorPeerGroupCalcRange
Class
→ WICBusinessObject

Fields
- _ID
- _InsertDt
- _InsertStfpID
- _MaxValue
- _MinValue
- _ModifyDt
- _ModifyStfpID
- _PG_ID
- _RangeTypeCd
- _StoreStructureCd

Properties
- ID
- InsertDt
- InsertStfpID
- MaxValue
- MinValue
- ModifyDt
- ModifyStfpID
- PG_ID
- RangeTypeCd
- StoreStructureCd

Methods
- New

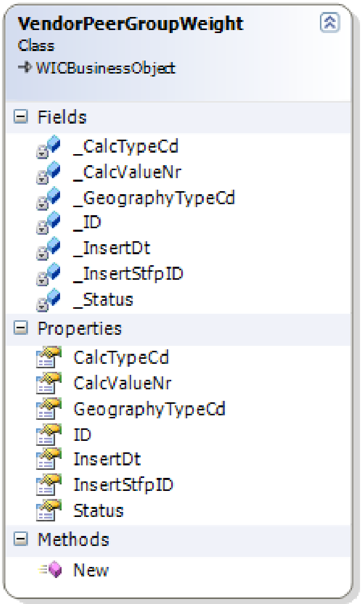


VendorPeerGroupGeography
Class
→ WICBusinessObject

Fields
- _CalcValueCd
- _GeographyTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _PG_ID
- _Status

Properties
- CalcValueCd
- GeographyTypeCd
- ID
- InsertDt
- InsertStfpID
- PG_ID
- Status

Methods
- New

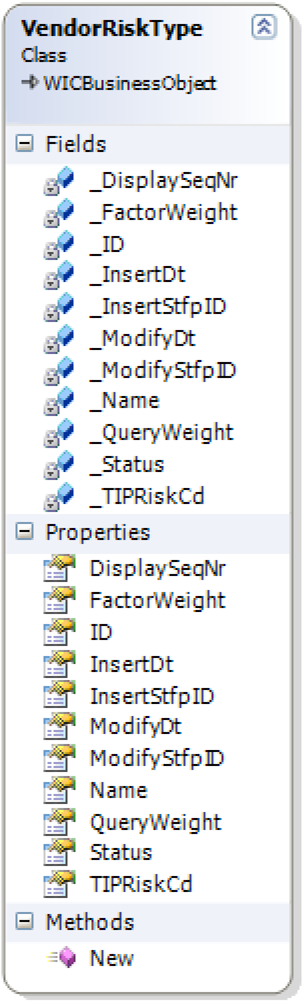


VendorPeerGroupWeight
Class
→ WICBusinessObject

Fields
- _CalcTypeCd
- _CalcValueNr
- _GeographyTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _Status

Properties
- CalcTypeCd
- CalcValueNr
- GeographyTypeCd
- ID
- InsertDt
- InsertStfpID
- Status

Methods
- New

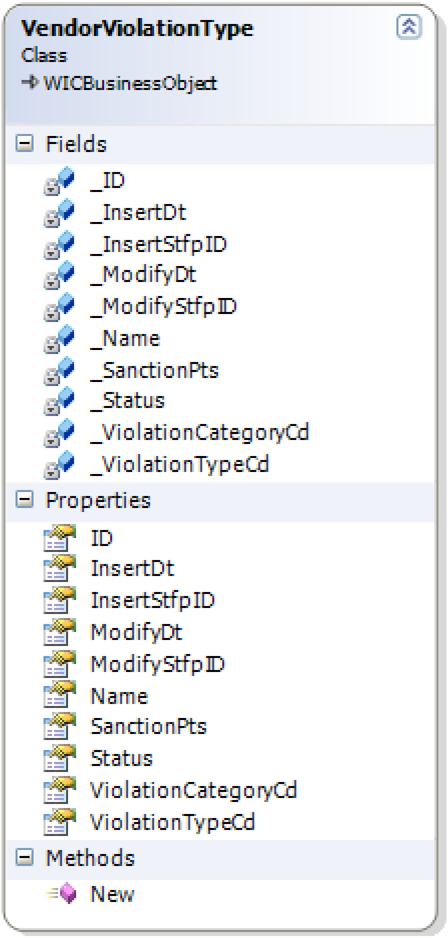


VendorRiskType
Class
→ WICBusinessObject

Fields
- _DisplaySeqNr
- _FactorWeight
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _QueryWeight
- _Status
- _TIPRiskCd

Properties
- DisplaySeqNr
- FactorWeight
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- QueryWeight
- Status
- TIPRiskCd

Methods
- New



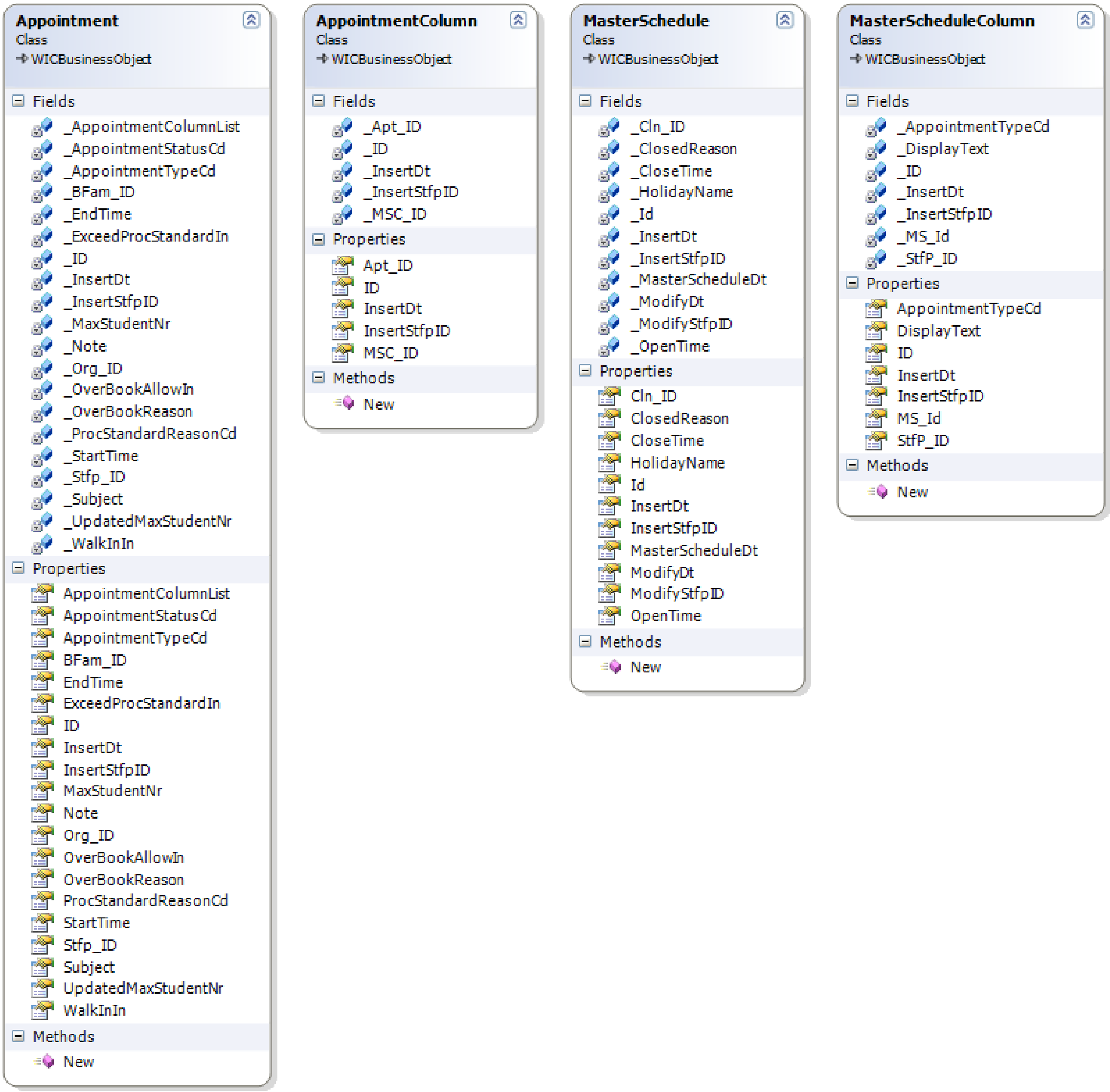
VendorViolationType
Class
→ WICBusinessObject

Fields
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _SanctionPts
- _Status
- _ViolationCategoryCd
- _ViolationTypeCd

Properties
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- SanctionPts
- Status
- ViolationCategoryCd
- ViolationTypeCd

Methods
- New

11.11 Scheduler



Appointment
Class
→ WICBusinessObject

Fields
- _AppointmentColumnList
- _AppointmentStatusCd
- _AppointmentTypeCd
- _BFam_ID
- _EndTime
- _ExceedProcStandardIn
- _ID
- _InsertDt
- _InsertStfpID
- _MaxStudentNr
- _Note
- _Org_ID
- _OverBookAllowIn
- _OverBookReason
- _ProcStandardReasonCd
- _StartTime
- _Stfp_ID
- _Subject
- _UpdatedMaxStudentNr
- _WalkInIn

Properties
- AppointmentColumnList
- AppointmentStatusCd
- AppointmentTypeCd
- BFam_ID
- EndTime
- ExceedProcStandardIn
- ID
- InsertDt
- InsertStfpID
- MaxStudentNr
- Note
- Org_ID
- OverBookAllowIn
- OverBookReason
- ProcStandardReasonCd
- StartTime
- Stfp_ID
- Subject
- UpdatedMaxStudentNr
- WalkInIn

Methods
- New

AppointmentColumn
Class
→ WICBusinessObject

Fields
- _Apt_ID
- _ID
- _InsertDt
- _InsertStfpID
- _MSC_ID

Properties
- Apt_ID
- ID
- InsertDt
- InsertStfpID
- MSC_ID

Methods
- New

MasterSchedule
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _ClosedReason
- _CloseTime
- _HolidayName
- _Id
- _InsertDt
- _InsertStfpID
- _MasterScheduleDt
- _ModifyDt
- _ModifyStfpID
- _OpenTime

Properties
- Cln_ID
- ClosedReason
- CloseTime
- HolidayName
- Id
- InsertDt
- InsertStfpID
- MasterScheduleDt
- ModifyDt
- ModifyStfpID
- OpenTime

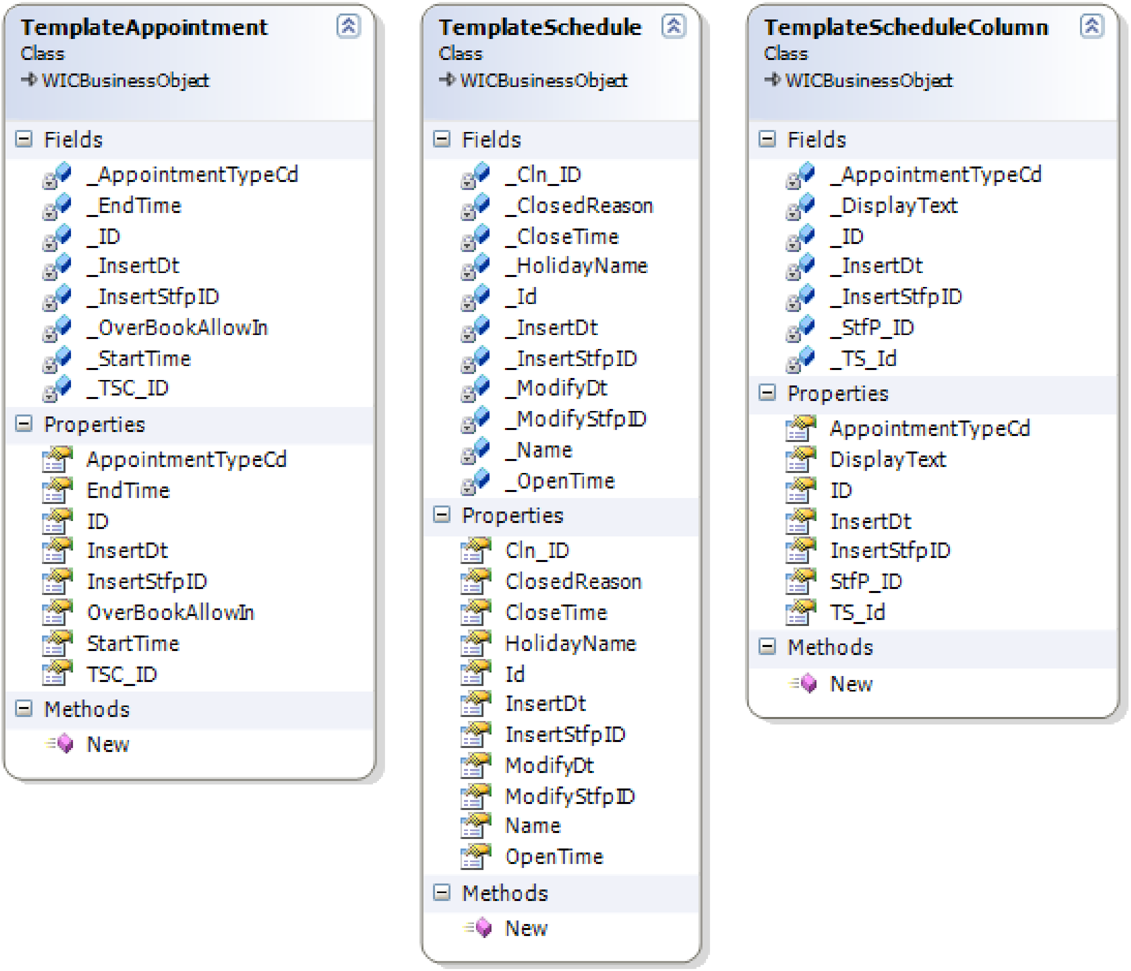
Methods
- New

MasterScheduleColumn
Class
→ WICBusinessObject

Fields
- _AppointmentTypeCd
- _DisplayText
- _ID
- _InsertDt
- _InsertStfpID
- _MS_Id
- _StfP_ID

Properties
- AppointmentTypeCd
- DisplayText
- ID
- InsertDt
- InsertStfpID
- MS_Id
- StfP_ID

Methods
- New



TemplateAppointment
Class
→ WICBusinessObject

Fields
- _AppointmentTypeCd
- _EndTime
- _ID
- _InsertDt
- _InsertStfpID
- _OverBookAllowIn
- _StartTime
- _TSC_ID

Properties
- AppointmentTypeCd
- EndTime
- ID
- InsertDt
- InsertStfpID
- OverBookAllowIn
- StartTime
- TSC_ID

Methods
- New

TemplateSchedule
Class
→ WICBusinessObject

Fields
- _Cln_ID
- _ClosedReason
- _CloseTime
- _HolidayName
- _Id
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _OpenTime

Properties
- Cln_ID
- ClosedReason
- CloseTime
- HolidayName
- Id
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- OpenTime

Methods
- New

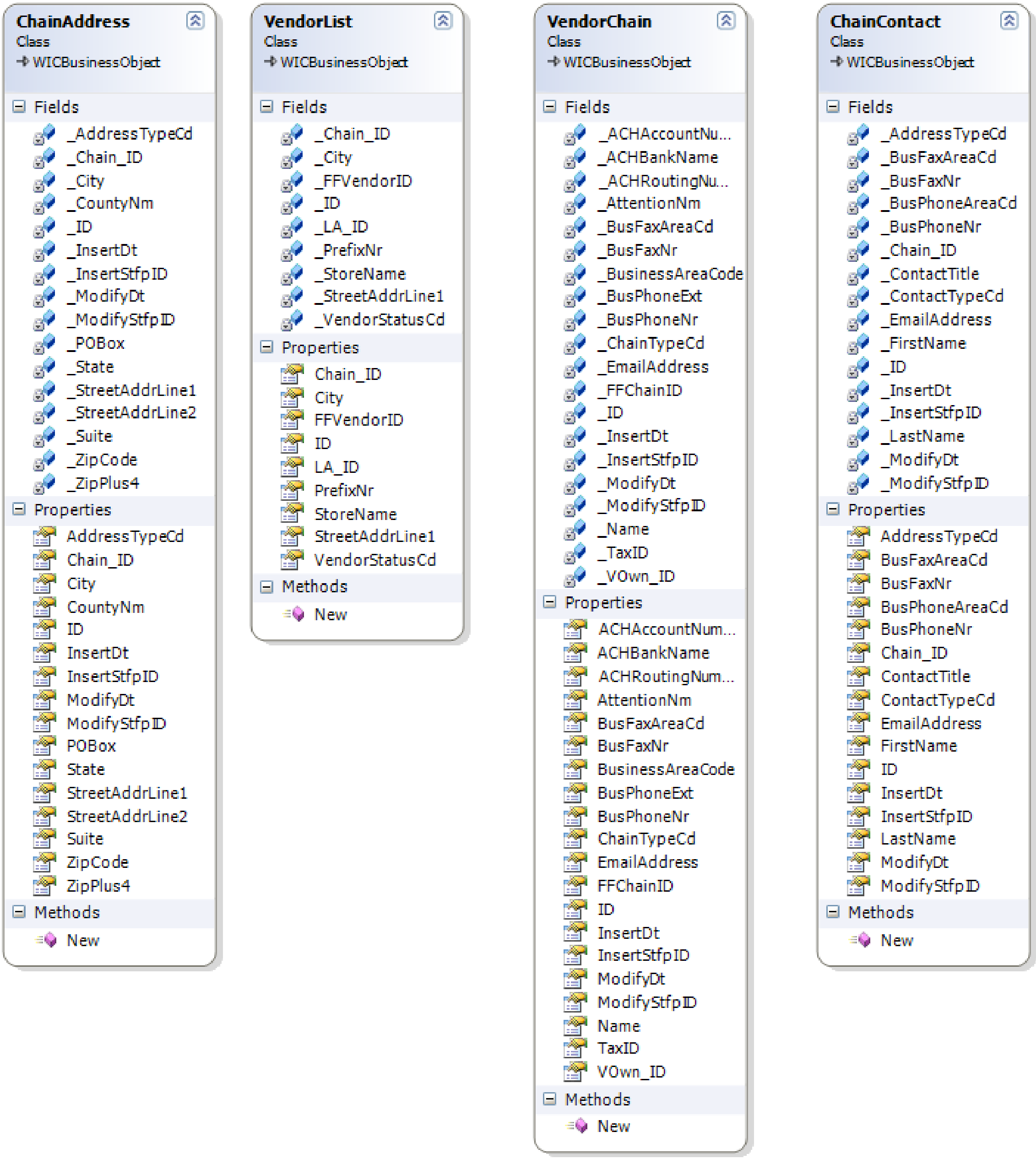
TemplateScheduleColumn
Class
→ WICBusinessObject

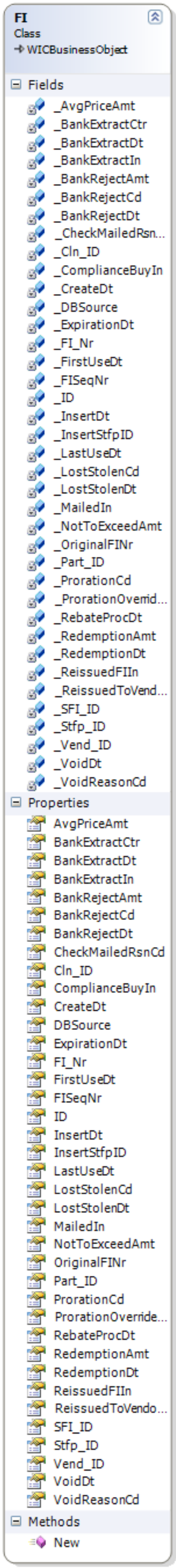
Fields
- _AppointmentTypeCd
- _DisplayText
- _ID
- _InsertDt
- _InsertStfpID
- _StfP_ID
- _TS_Id

Properties
- AppointmentTypeCd
- DisplayText
- ID
- InsertDt
- InsertStfpID
- StfP_ID
- TS_Id

Methods
- New

11.12 Vendor Management



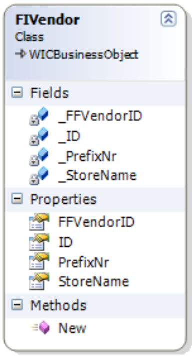


FI
Class
→ WICBusinessObject

Fields
_AvgPriceAmt
_BankExtractCtr
_BankExtractDt
_BankExtractIn
_BankRejectAmt
_BankRejectCd
_BankRejectDt
_CheckMailedRsn...
_Cln_ID
_ComplianceBuyIn
_CreateDt
_DBSource
_ExpirationDt
_FI_Nr
_FirstUseDt
_FISeqNr
_ID
_InsertDt
_InsertStfpID
_LastUseDt
_LostStolenCd
_LostStolenDt
_MailedIn
_NotToExceedAmt
_OriginalFINr
_Part_ID
_ProrationCd
_ProrationOverrid...
_RebateProcDt
_RedemptionAmt
_RedemptionDt
_ReissuedFIIn
_ReissuedToVend...
_SFI_ID
_Stfp_ID
_Vend_ID
_VoidDt
_VoidReasonCd

Properties
AvgPriceAmt
BankExtractCtr
BankExtractDt
BankExtractIn
BankRejectAmt
BankRejectCd
BankRejectDt
CheckMailedRsnCd
Cln_ID
ComplianceBuyIn
CreateDt
DBSource
ExpirationDt
FI_Nr
FirstUseDt
FISeqNr
ID
InsertDt
InsertStfpID
LastUseDt
LostStolenCd
LostStolenDt
MailedIn
NotToExceedAmt
OriginalFINr
Part_ID
ProrationCd
ProrationOverride...
RebateProcDt
RedemptionAmt
RedemptionDt
ReissuedFIIn
ReissuedToVendo...
SFI_ID
Stfp_ID
Vend_ID
VoidDt
VoidReasonCd

Methods
New

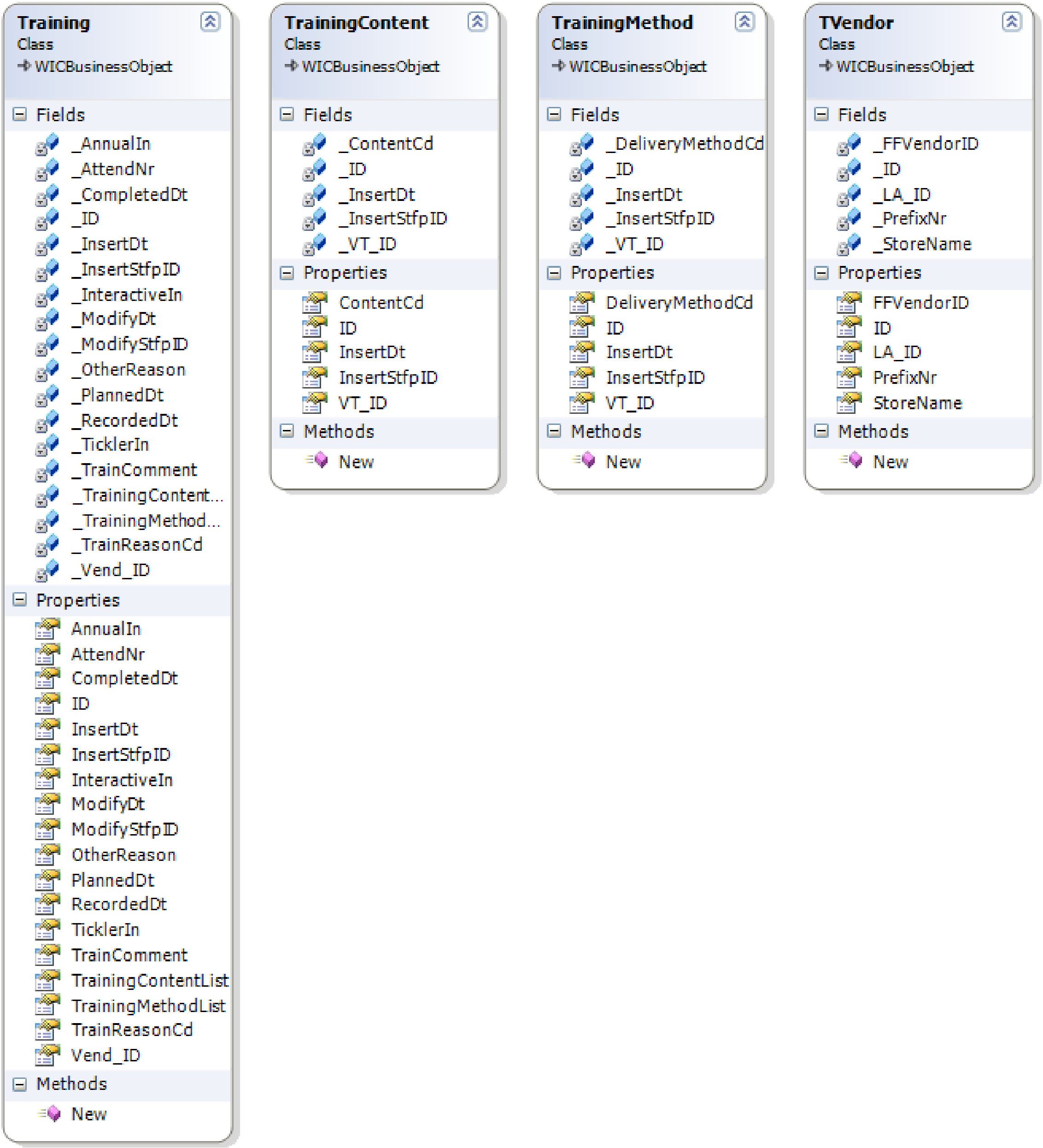


FIVendor
Class
→ WICBusinessObject

Fields
_FFVendorID
_ID
_PrefixNr
_StoreName

Properties
FFVendorID
ID
PrefixNr
StoreName

Methods
New



Training
Class
WICBusinessObject

Fields
- _AnnualIn
- _AttendNr
- _CompletedDt
- _ID
- _InsertDt
- _InsertStfpID
- _InteractiveIn
- _ModifyDt
- _ModifyStfpID
- _OtherReason
- _PlannedDt
- _RecordedDt
- _TicklerIn
- _TrainComment
- _TrainingContent...
- _TrainingMethod...
- _TrainReasonCd
- _Vend_ID

Properties
- AnnualIn
- AttendNr
- CompletedDt
- ID
- InsertDt
- InsertStfpID
- InteractiveIn
- ModifyDt
- ModifyStfpID
- OtherReason
- PlannedDt
- RecordedDt
- TicklerIn
- TrainComment
- TrainingContentList
- TrainingMethodList
- TrainReasonCd
- Vend_ID

Methods
- New

TrainingContent
Class
WICBusinessObject

Fields
- _ContentCd
- _ID
- _InsertDt
- _InsertStfpID
- _VT_ID

Properties
- ContentCd
- ID
- InsertDt
- InsertStfpID
- VT_ID

Methods
- New

TrainingMethod
Class
WICBusinessObject

Fields
- _DeliveryMethodCd
- _ID
- _InsertDt
- _InsertStfpID
- _VT_ID

Properties
- DeliveryMethodCd
- ID
- InsertDt
- InsertStfpID
- VT_ID

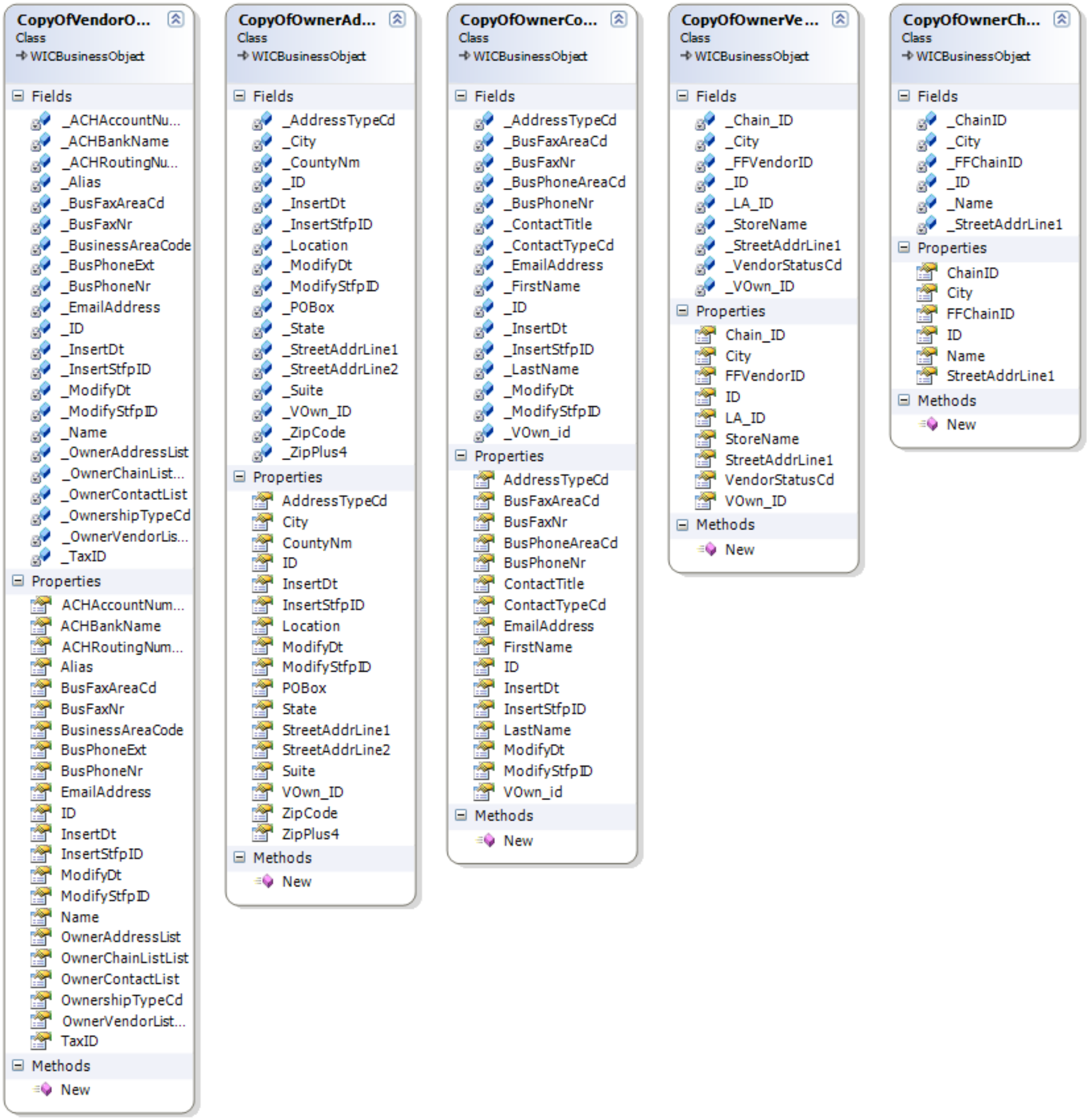
Methods
- New

TVendor
Class
WICBusinessObject

Fields
- _FFVendorID
- _ID
- _LA_ID
- _PrefixNr
- _StoreName

Properties
- FFVendorID
- ID
- LA_ID
- PrefixNr
- StoreName

Methods
- New



CopyOfVendorO...
Class
→ WICBusinessObject

Fields
- _ACHAccountNu...
- _ACHBankName
- _ACHRoutingNu...
- _Alias
- _BusFaxAreaCd
- _BusFaxNr
- _BusinessAreaCode
- _BusPhoneExt
- _BusPhoneNr
- _EmailAddress
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _OwnerAddressList
- _OwnerChainList...
- _OwnerContactList
- _OwnershipTypeCd
- _OwnerVendorLis...
- _TaxID

Properties
- ACHAccountNum...
- ACHBankName
- ACHRoutingNum...
- Alias
- BusFaxAreaCd
- BusFaxNr
- BusinessAreaCode
- BusPhoneExt
- BusPhoneNr
- EmailAddress
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- OwnerAddressList
- OwnerChainListList
- OwnerContactList
- OwnershipTypeCd
- OwnerVendorList...
- TaxID

Methods
- New

CopyOfOwnerAd...
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _City
- _CountyNm
- _ID
- _InsertDt
- _InsertStfpID
- _Location
- _ModifyDt
- _ModifyStfpID
- _POBox
- _State
- _StreetAddrLine1
- _StreetAddrLine2
- _Suite
- _VOwn_ID
- _ZipCode
- _ZipPlus4

Properties
- AddressTypeCd
- City
- CountyNm
- ID
- InsertDt
- InsertStfpID
- Location
- ModifyDt
- ModifyStfpID
- POBox
- State
- StreetAddrLine1
- StreetAddrLine2
- Suite
- VOwn_ID
- ZipCode
- ZipPlus4

Methods
- New

CopyOfOwnerCo...
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _BusFaxAreaCd
- _BusFaxNr
- _BusPhoneAreaCd
- _BusPhoneNr
- _ContactTitle
- _ContactTypeCd
- _EmailAddress
- _FirstName
- _ID
- _InsertDt
- _InsertStfpID
- _LastName
- _ModifyDt
- _ModifyStfpID
- _VOwn_id

Properties
- AddressTypeCd
- BusFaxAreaCd
- BusFaxNr
- BusPhoneAreaCd
- BusPhoneNr
- ContactTitle
- ContactTypeCd
- EmailAddress
- FirstName
- ID
- InsertDt
- InsertStfpID
- LastName
- ModifyDt
- ModifyStfpID
- VOwn_id

Methods
- New

CopyOfOwnerVe...
Class
→ WICBusinessObject

Fields
- _Chain_ID
- _City
- _FFVendorID
- _ID
- _LA_ID
- _StoreName
- _StreetAddrLine1
- _VendorStatusCd
- _VOwn_ID

Properties
- Chain_ID
- City
- FFVendorID
- ID
- LA_ID
- StoreName
- StreetAddrLine1
- VendorStatusCd
- VOwn_ID

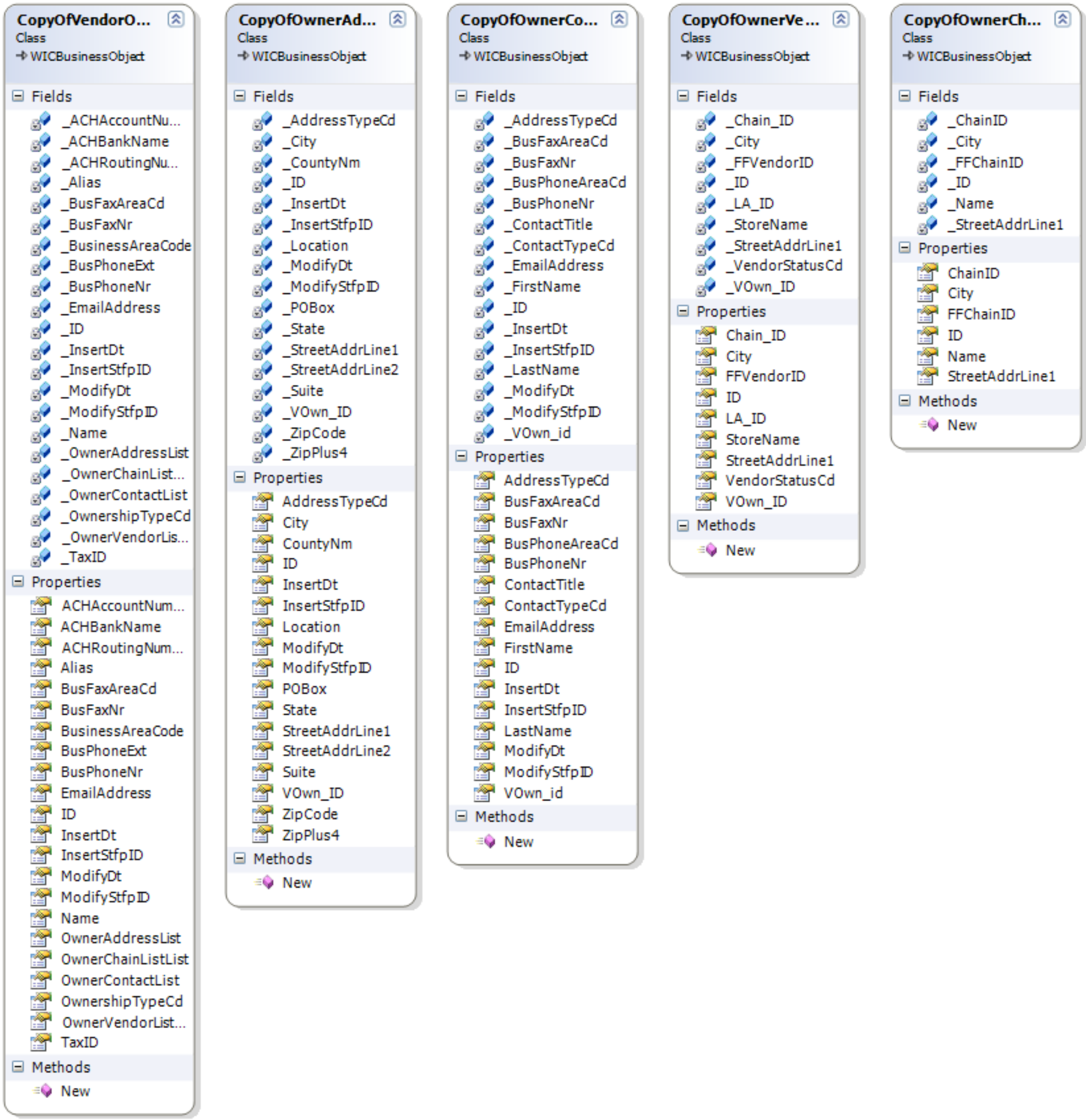
Methods
- New

CopyOfOwnerCh...
Class
→ WICBusinessObject

Fields
- _ChainID
- _City
- _FFChainID
- _ID
- _Name
- _StreetAddrLine1

Properties
- ChainID
- City
- FFChainID
- ID
- Name
- StreetAddrLine1

Methods
- New



CopyOfVendorO...
Class
→ WICBusinessObject

Fields
- _ACHAccountNu...
- _ACHBankName
- _ACHRoutingNu...
- _Alias
- _BusFaxAreaCd
- _BusFaxNr
- _BusinessAreaCode
- _BusPhoneExt
- _BusPhoneNr
- _EmailAddress
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Name
- _OwnerAddressList
- _OwnerChainList...
- _OwnerContactList
- _OwnershipTypeCd
- _OwnerVendorLis...
- _TaxID

Properties
- ACHAccountNum...
- ACHBankName
- ACHRoutingNum...
- Alias
- BusFaxAreaCd
- BusFaxNr
- BusinessAreaCode
- BusPhoneExt
- BusPhoneNr
- EmailAddress
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Name
- OwnerAddressList
- OwnerChainListList
- OwnerContactList
- OwnershipTypeCd
- OwnerVendorList...
- TaxID

Methods
- New

CopyOfOwnerAd...
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _City
- _CountyNm
- _ID
- _InsertDt
- _InsertStfpID
- _Location
- _ModifyDt
- _ModifyStfpID
- _POBox
- _State
- _StreetAddrLine1
- _StreetAddrLine2
- _Suite
- _VOwn_ID
- _ZipCode
- _ZipPlus4

Properties
- AddressTypeCd
- City
- CountyNm
- ID
- InsertDt
- InsertStfpID
- Location
- ModifyDt
- ModifyStfpID
- POBox
- State
- StreetAddrLine1
- StreetAddrLine2
- Suite
- VOwn_ID
- ZipCode
- ZipPlus4

Methods
- New

CopyOfOwnerCo...
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _BusFaxAreaCd
- _BusFaxNr
- _BusPhoneAreaCd
- _BusPhoneNr
- _ContactTitle
- _ContactTypeCd
- _EmailAddress
- _FirstName
- _ID
- _InsertDt
- _InsertStfpID
- _LastName
- _ModifyDt
- _ModifyStfpID
- _VOwn_id

Properties
- AddressTypeCd
- BusFaxAreaCd
- BusFaxNr
- BusPhoneAreaCd
- BusPhoneNr
- ContactTitle
- ContactTypeCd
- EmailAddress
- FirstName
- ID
- InsertDt
- InsertStfpID
- LastName
- ModifyDt
- ModifyStfpID
- VOwn_id

Methods
- New

CopyOfOwnerVe...
Class
→ WICBusinessObject

Fields
- _Chain_ID
- _City
- _FFVendorID
- _ID
- _LA_ID
- _StoreName
- _StreetAddrLine1
- _VendorStatusCd
- _VOwn_ID

Properties
- Chain_ID
- City
- FFVendorID
- ID
- LA_ID
- StoreName
- StreetAddrLine1
- VendorStatusCd
- VOwn_ID

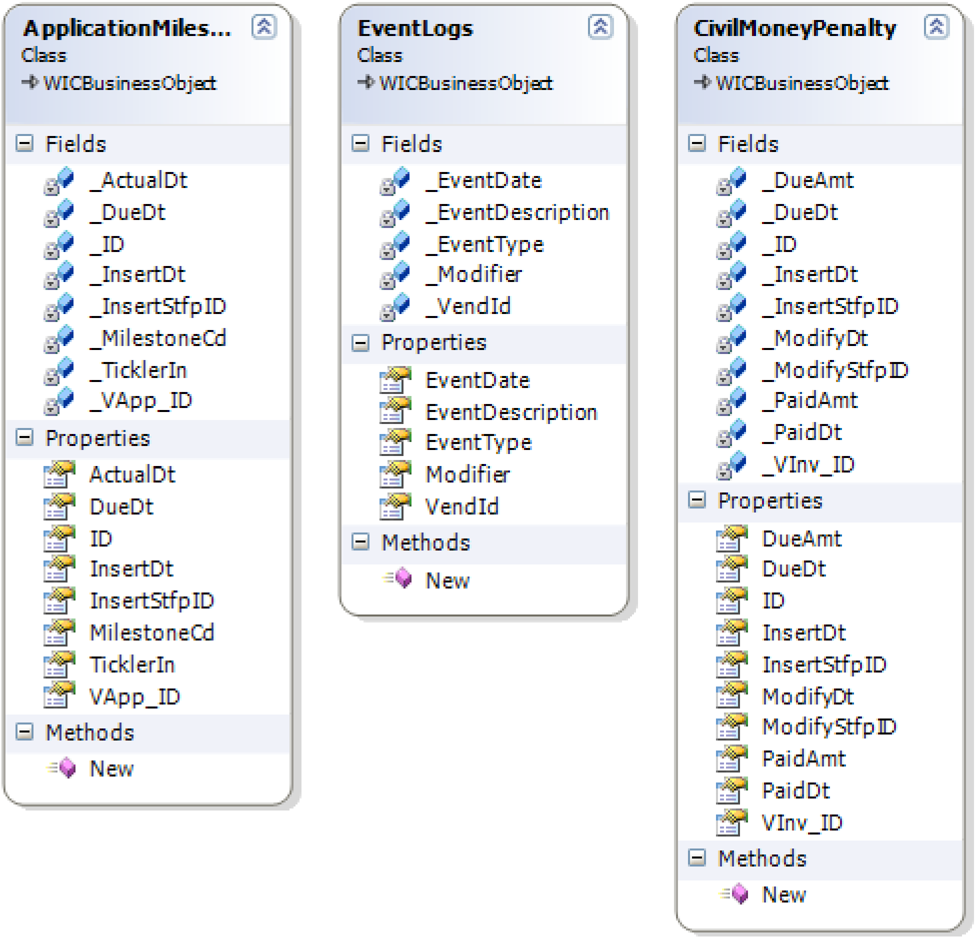
Methods
- New

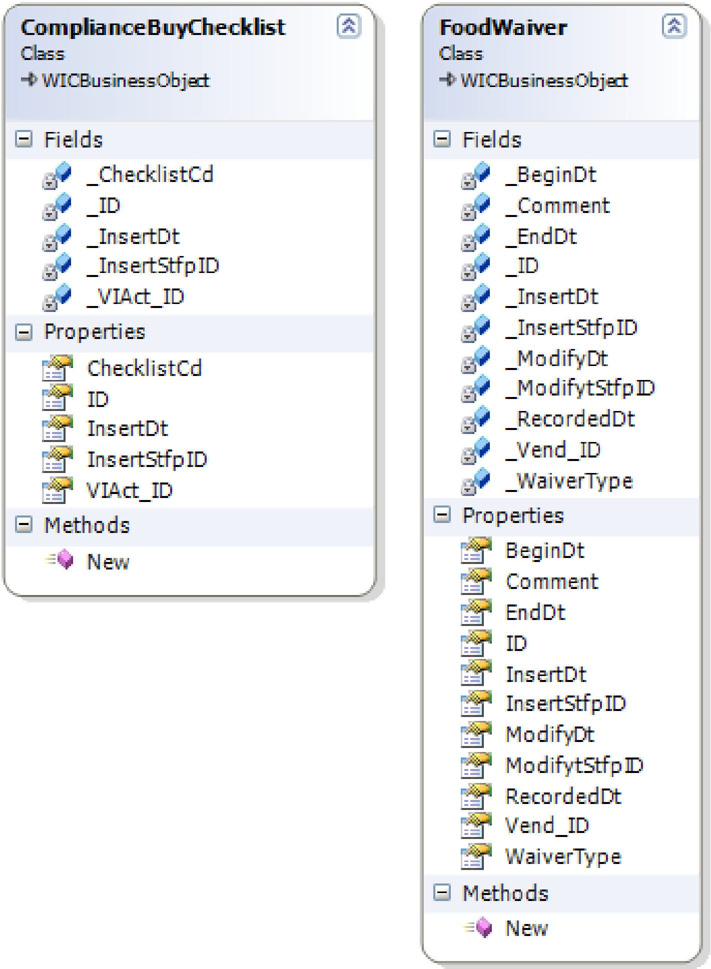
CopyOfOwnerCh...
Class
→ WICBusinessObject

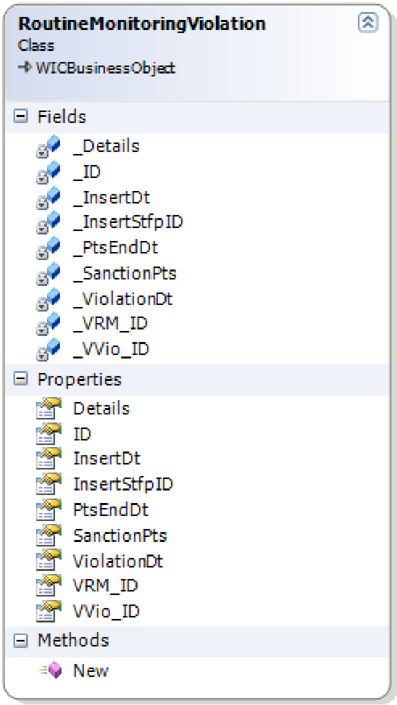
Fields
- _ChainID
- _City
- _FFChainID
- _ID
- _Name
- _StreetAddrLine1

Properties
- ChainID
- City
- FFChainID
- ID
- Name
- StreetAddrLine1

Methods
- New


ApplicationMiles...
Class
WICBusinessObject
Fields
_ActualDt
_DueDt
_ID
_InsertDt
_InsertStfpID
_MilestoneCd
_TicklerIn
_VApp_ID
Properties
ActualDt
DueDt
ID
InsertDt
InsertStfpID
MilestoneCd
TicklerIn
VApp_ID
Methods
New
EventLogs
Class
WICBusinessObject
Fields
_EventDate
_EventDescription
_EventType
_Modifier
_VendId
Properties
EventDate
EventDescription
EventType
Modifier
VendId
Methods
New
CivilMoneyPenalty
Class
WICBusinessObject
Fields
_DueAmt
_DueDt
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_PaidAmt
_PaidDt
_VInv_ID
Properties
DueAmt
DueDt
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
PaidAmt
PaidDt
VInv_ID
Methods
New


ComplianceBuyChecklist
Class
WICBusinessObject
Fields
_ChecklistCd
_ID
_InsertDt
_InsertStfpID
_VIAct_ID
Properties
ChecklistCd
ID
InsertDt
InsertStfpID
VIAct_ID
Methods
New
FoodWaiver
Class
WICBusinessObject
Fields
_BeginDt
_Comment
_EndDt
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifytStfpID
_RecordedDt
_Vend_ID
_WaiverType
Properties
BeginDt
Comment
EndDt
ID
InsertDt
InsertStfpID
ModifyDt
ModifytStfpID
RecordedDt
Vend_ID
WaiverType
Methods
New
LimitingCriteriaChecklist
Class
WICBusinessObject
Fields
_ChecklistCd
_ID
_InsertDt
_InsertStfpID
_VApp_ID
Properties
ChecklistCd
ID
InsertDt
InsertStfpID
VApp_ID
Methods
New

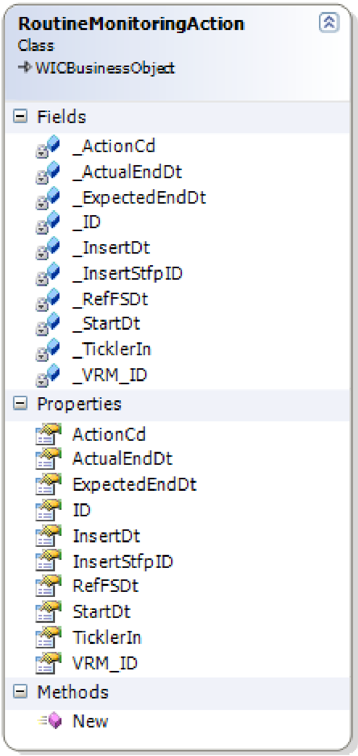


RoutineMonitoringViolation
Class
→ WICBusinessObject

Fields
- _Details
- _ID
- _InsertDt
- _InsertStfpID
- _PtsEndDt
- _SanctionPts
- _ViolationDt
- _VRM_ID
- _VVio_ID

Properties
- Details
- ID
- InsertDt
- InsertStfpID
- PtsEndDt
- SanctionPts
- ViolationDt
- VRM_ID
- VVio_ID

Methods
- New

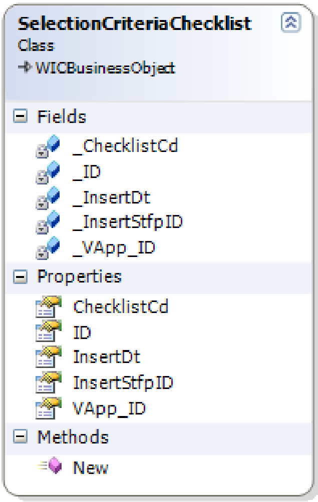


RoutineMonitoringAction
Class
→ WICBusinessObject

Fields
- _ActionCd
- _ActualEndDt
- _ExpectedEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _RefFSDt
- _StartDt
- _TicklerIn
- _VRM_ID

Properties
- ActionCd
- ActualEndDt
- ExpectedEndDt
- ID
- InsertDt
- InsertStfpID
- RefFSDt
- StartDt
- TicklerIn
- VRM_ID

Methods
- New

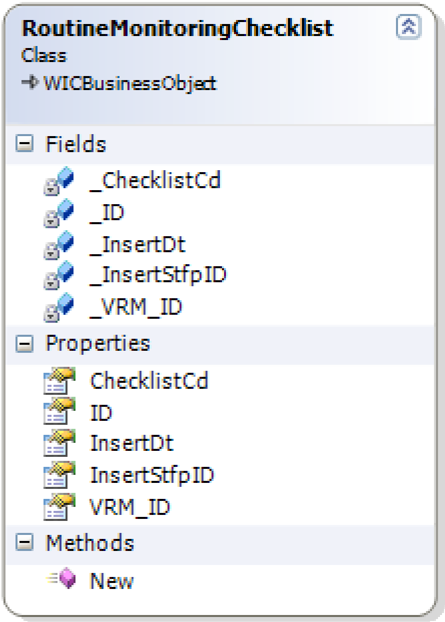


SelectionCriteriaChecklist
Class
→ WICBusinessObject

Fields
- _ChecklistCd
- _ID
- _InsertDt
- _InsertStfpID
- _VApp_ID

Properties
- ChecklistCd
- ID
- InsertDt
- InsertStfpID
- VApp_ID

Methods
- New

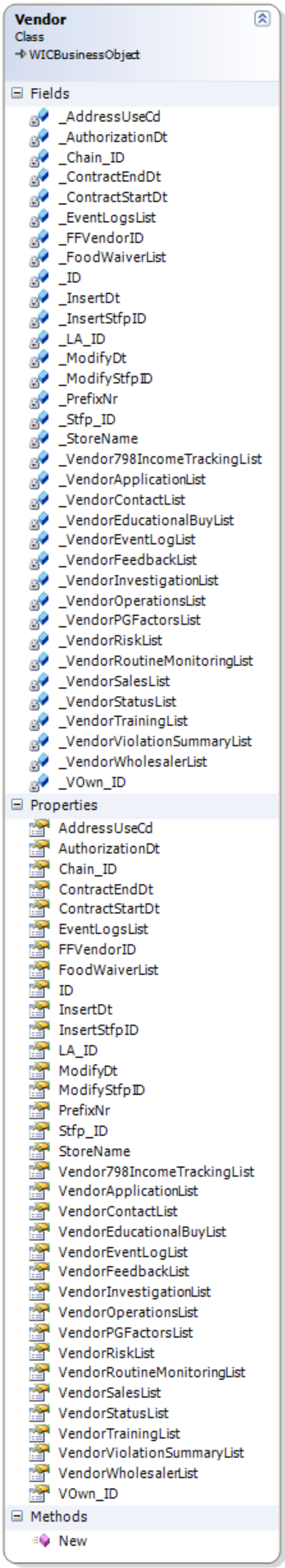


RoutineMonitoringChecklist
Class
→ WICBusinessObject

Fields
- _ChecklistCd
- _ID
- _InsertDt
- _InsertStfpID
- _VRM_ID

Properties
- ChecklistCd
- ID
- InsertDt
- InsertStfpID
- VRM_ID

Methods
- New

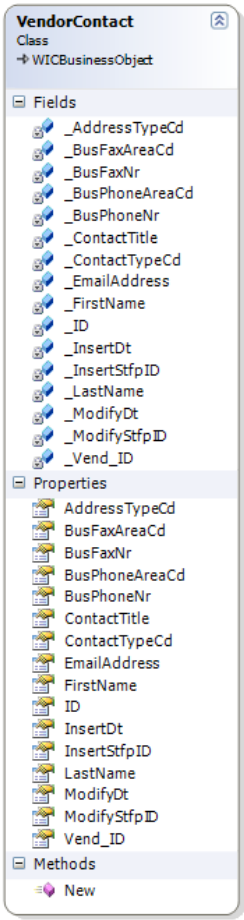


Vendor
Class
→ WICBusinessObject

Fields
- _AddressUseCd
- _AuthorizationDt
- _Chain_ID
- _ContractEndDt
- _ContractStartDt
- _EventLogsList
- _FFVendorID
- _FoodWaiverList
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _PrefixNr
- _Stfp_ID
- _StoreName
- _Vendor798IncomeTrackingList
- _VendorApplicationList
- _VendorContactList
- _VendorEducationalBuyList
- _VendorEventLogList
- _VendorFeedbackList
- _VendorInvestigationList
- _VendorOperationsList
- _VendorPGFactorsList
- _VendorRiskList
- _VendorRoutineMonitoringList
- _VendorSalesList
- _VendorStatusList
- _VendorTrainingList
- _VendorViolationSummaryList
- _VendorWholesalerList
- _VOwn_ID

Properties
- AddressUseCd
- AuthorizationDt
- Chain_ID
- ContractEndDt
- ContractStartDt
- EventLogsList
- FFVendorID
- FoodWaiverList
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- PrefixNr
- Stfp_ID
- StoreName
- Vendor798IncomeTrackingList
- VendorApplicationList
- VendorContactList
- VendorEducationalBuyList
- VendorEventLogList
- VendorFeedbackList
- VendorInvestigationList
- VendorOperationsList
- VendorPGFactorsList
- VendorRiskList
- VendorRoutineMonitoringList
- VendorSalesList
- VendorStatusList
- VendorTrainingList
- VendorViolationSummaryList
- VendorWholesalerList
- VOwn_ID

Methods
- New

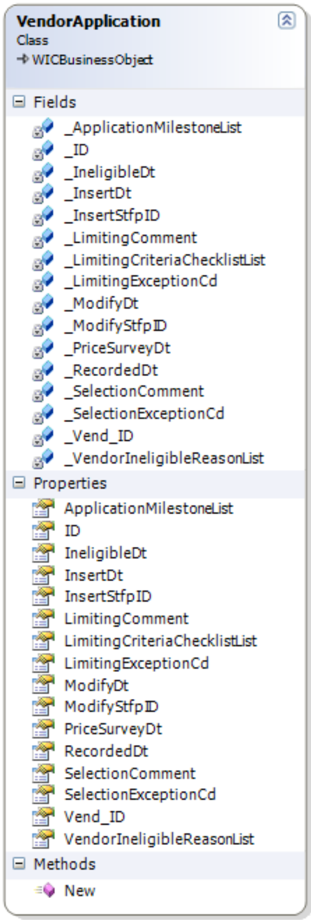


VendorContact
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _BusFaxAreaCd
- _BusFaxNr
- _BusPhoneAreaCd
- _BusPhoneNr
- _ContactTitle
- _ContactTypeCd
- _EmailAddress
- _FirstName
- _ID
- _InsertDt
- _InsertStfpID
- _LastName
- _ModifyDt
- _ModifyStfpID
- _Vend_ID

Properties
- AddressTypeCd
- BusFaxAreaCd
- BusFaxNr
- BusPhoneAreaCd
- BusPhoneNr
- ContactTitle
- ContactTypeCd
- EmailAddress
- FirstName
- ID
- InsertDt
- InsertStfpID
- LastName
- ModifyDt
- ModifyStfpID
- Vend_ID

Methods
- New

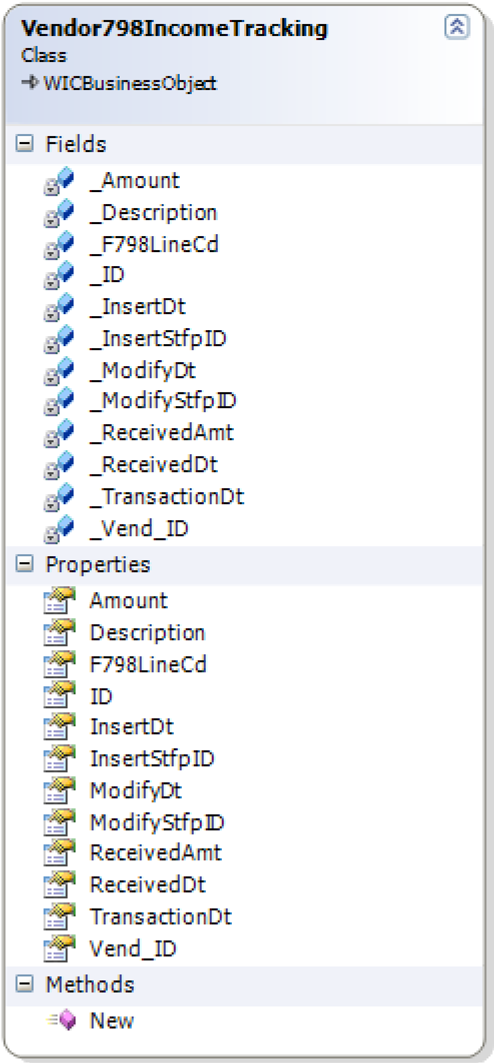


VendorApplication
Class
→ WICBusinessObject

Fields
- _ApplicationMilestoneList
- _ID
- _IneligibleDt
- _InsertDt
- _InsertStfpID
- _LimitingComment
- _LimitingCriteriaChecklistList
- _LimitingExceptionCd
- _ModifyDt
- _ModifyStfpID
- _PriceSurveyDt
- _RecordedDt
- _SelectionComment
- _SelectionExceptionCd
- _Vend_ID
- _VendorIneligibleReasonList

Properties
- ApplicationMilestoneList
- ID
- IneligibleDt
- InsertDt
- InsertStfpID
- LimitingComment
- LimitingCriteriaChecklistList
- LimitingExceptionCd
- ModifyDt
- ModifyStfpID
- PriceSurveyDt
- RecordedDt
- SelectionComment
- SelectionExceptionCd
- Vend_ID
- VendorIneligibleReasonList

Methods
- New

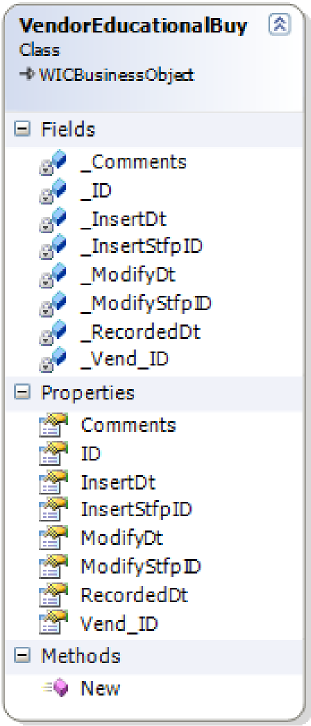


Vendor798IncomeTracking
Class
→ WICBusinessObject

Fields
- _Amount
- _Description
- _F798LineCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _ReceivedAmt
- _ReceivedDt
- _TransactionDt
- _Vend_ID

Properties
- Amount
- Description
- F798LineCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- ReceivedAmt
- ReceivedDt
- TransactionDt
- Vend_ID

Methods
- New

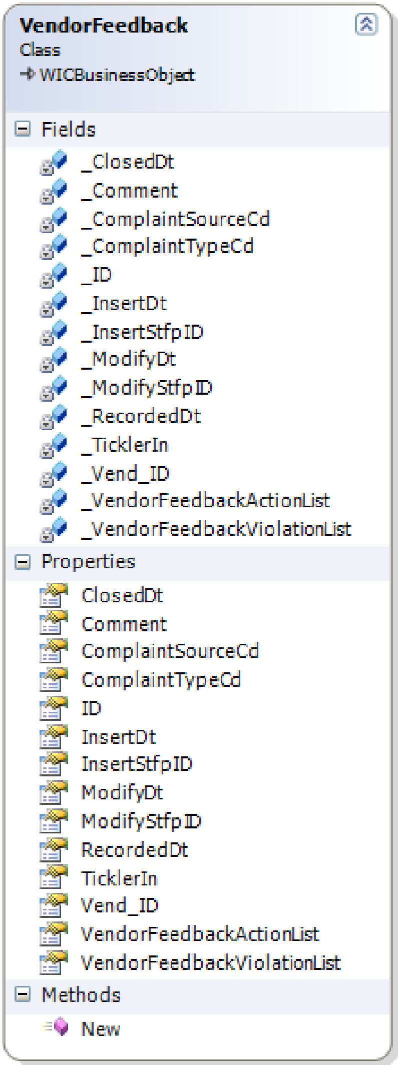


VendorEducationalBuy
Class
→ WICBusinessObject

⊟ Fields
- _Comments
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _Vend_ID

⊟ Properties
- Comments
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- Vend_ID

⊟ Methods
- New

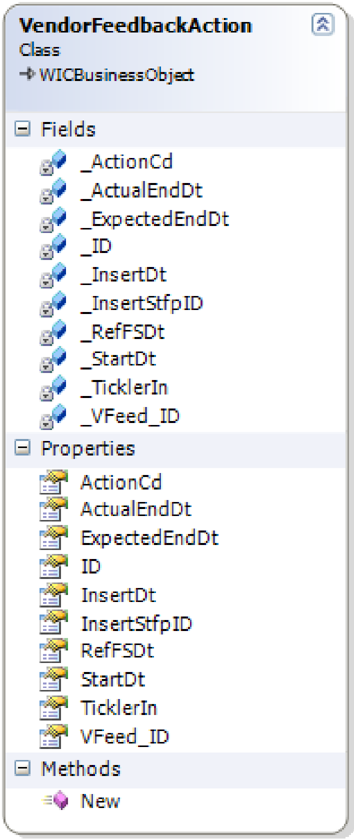


VendorFeedback
Class
→ WICBusinessObject

⊟ Fields
- _ClosedDt
- _Comment
- _ComplaintSourceCd
- _ComplaintTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _TicklerIn
- _Vend_ID
- _VendorFeedbackActionList
- _VendorFeedbackViolationList

⊟ Properties
- ClosedDt
- Comment
- ComplaintSourceCd
- ComplaintTypeCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- TicklerIn
- Vend_ID
- VendorFeedbackActionList
- VendorFeedbackViolationList

⊟ Methods
- New

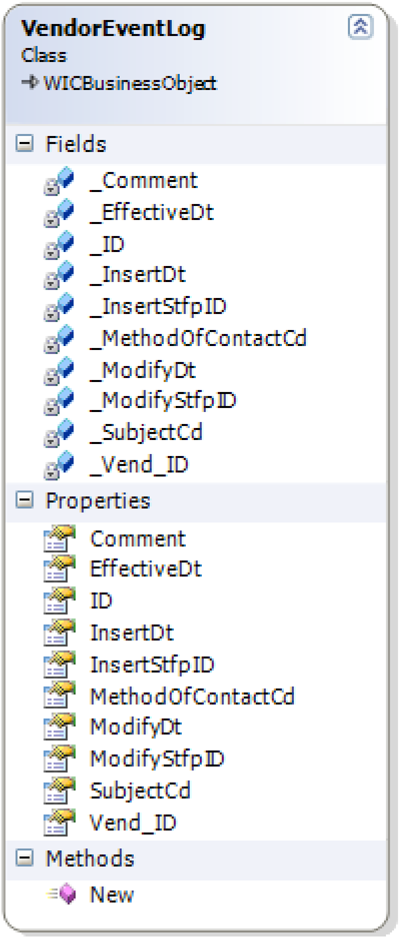


VendorFeedbackAction
Class
→ WICBusinessObject

⊟ Fields
- _ActionCd
- _ActualEndDt
- _ExpectedEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _RefFSDt
- _StartDt
- _TicklerIn
- _VFeed_ID

⊟ Properties
- ActionCd
- ActualEndDt
- ExpectedEndDt
- ID
- InsertDt
- InsertStfpID
- RefFSDt
- StartDt
- TicklerIn
- VFeed_ID

⊟ Methods
- New

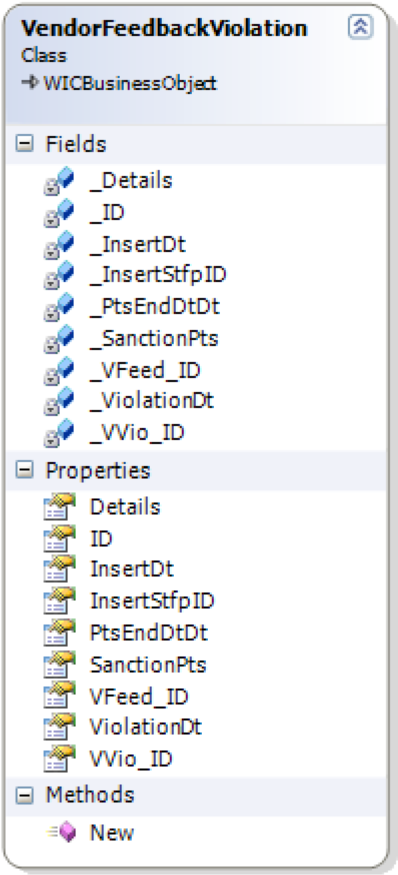


VendorEventLog
Class
→ WICBusinessObject

Fields
- _Comment
- _EffectiveDt
- _ID
- _InsertDt
- _InsertStfpID
- _MethodOfContactCd
- _ModifyDt
- _ModifyStfpID
- _SubjectCd
- _Vend_ID

Properties
- Comment
- EffectiveDt
- ID
- InsertDt
- InsertStfpID
- MethodOfContactCd
- ModifyDt
- ModifyStfpID
- SubjectCd
- Vend_ID

Methods
- New

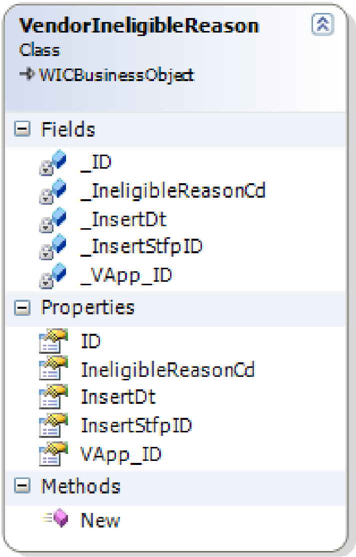


VendorFeedbackViolation
Class
→ WICBusinessObject

Fields
- _Details
- _ID
- _InsertDt
- _InsertStfpID
- _PtsEndDtDt
- _SanctionPts
- _VFeed_ID
- _ViolationDt
- _VVio_ID

Properties
- Details
- ID
- InsertDt
- InsertStfpID
- PtsEndDtDt
- SanctionPts
- VFeed_ID
- ViolationDt
- VVio_ID

Methods
- New

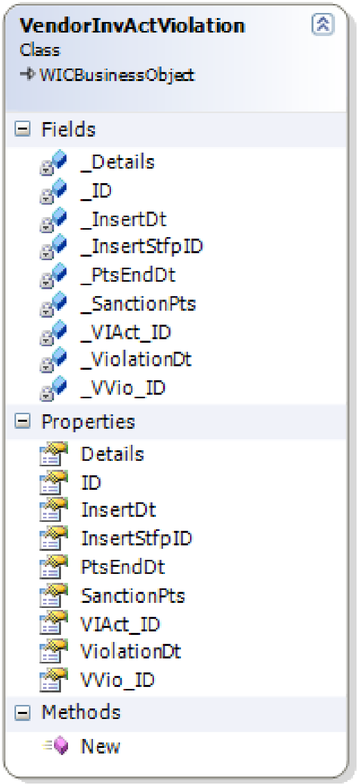


VendorIneligibleReason
Class
→ WICBusinessObject

Fields
- _ID
- _IneligibleReasonCd
- _InsertDt
- _InsertStfpID
- _VApp_ID

Properties
- ID
- IneligibleReasonCd
- InsertDt
- InsertStfpID
- VApp_ID

Methods
- New

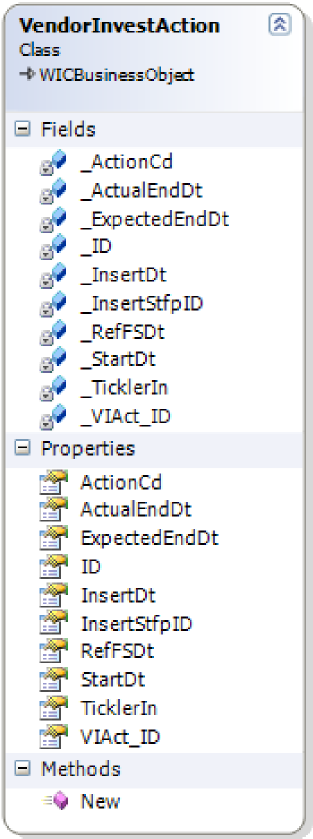


VendorInvActViolation
Class
→ WICBusinessObject

Fields
- _Details
- _ID
- _InsertDt
- _InsertStfpID
- _PtsEndDt
- _SanctionPts
- _VIAct_ID
- _ViolationDt
- _VVio_ID

Properties
- Details
- ID
- InsertDt
- InsertStfpID
- PtsEndDt
- SanctionPts
- VIAct_ID
- ViolationDt
- VVio_ID

Methods
- New

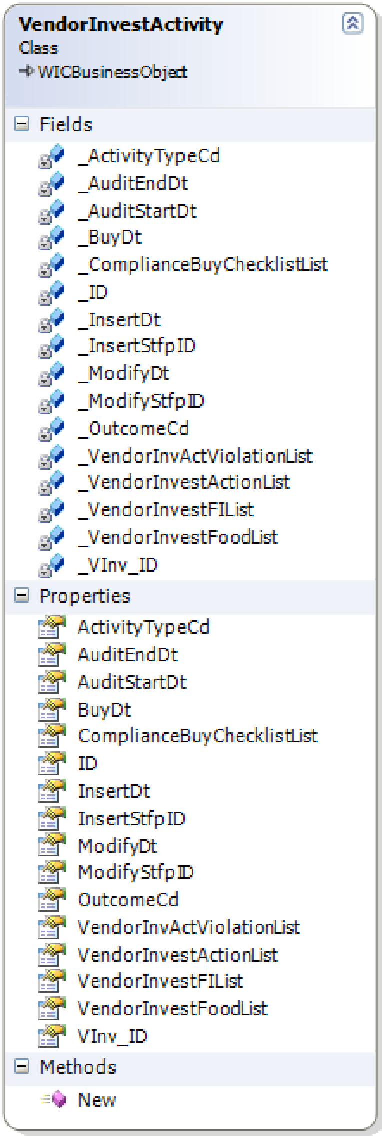


VendorInvestAction
Class
→ WICBusinessObject

Fields
- _ActionCd
- _ActualEndDt
- _ExpectedEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _RefFSDt
- _StartDt
- _TicklerIn
- _VIAct_ID

Properties
- ActionCd
- ActualEndDt
- ExpectedEndDt
- ID
- InsertDt
- InsertStfpID
- RefFSDt
- StartDt
- TicklerIn
- VIAct_ID

Methods
- New

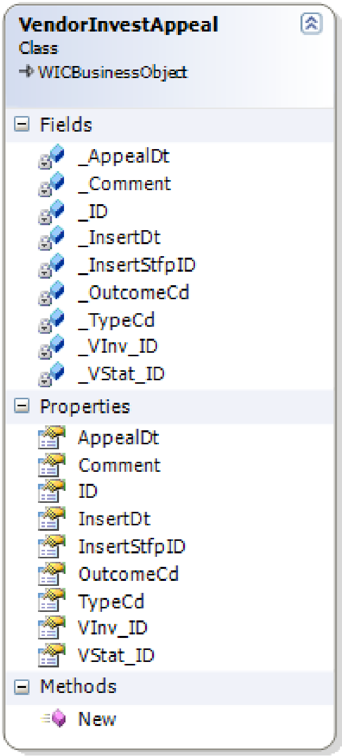


VendorInvestActivity
Class
→ WICBusinessObject

Fields
- _ActivityTypeCd
- _AuditEndDt
- _AuditStartDt
- _BuyDt
- _ComplianceBuyChecklistList
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OutcomeCd
- _VendorInvActViolationList
- _VendorInvestActionList
- _VendorInvestFIList
- _VendorInvestFoodList
- _VInv_ID

Properties
- ActivityTypeCd
- AuditEndDt
- AuditStartDt
- BuyDt
- ComplianceBuyChecklistList
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OutcomeCd
- VendorInvActViolationList
- VendorInvestActionList
- VendorInvestFIList
- VendorInvestFoodList
- VInv_ID

Methods
- New

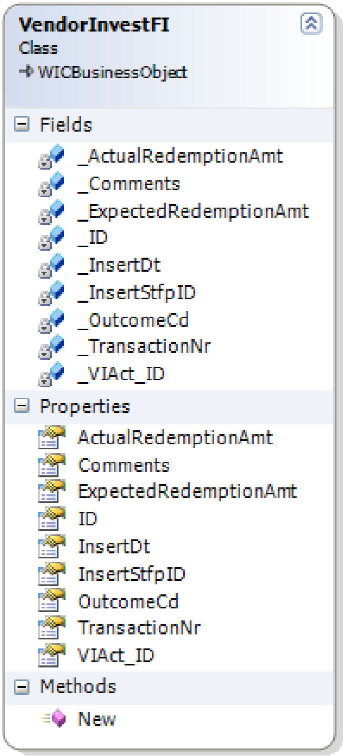


VendorInvestAppeal
Class
→ WICBusinessObject

Fields
- _AppealDt
- _Comment
- _ID
- _InsertDt
- _InsertStfpID
- _OutcomeCd
- _TypeCd
- _VInv_ID
- _VStat_ID

Properties
- AppealDt
- Comment
- ID
- InsertDt
- InsertStfpID
- OutcomeCd
- TypeCd
- VInv_ID
- VStat_ID

Methods
- New

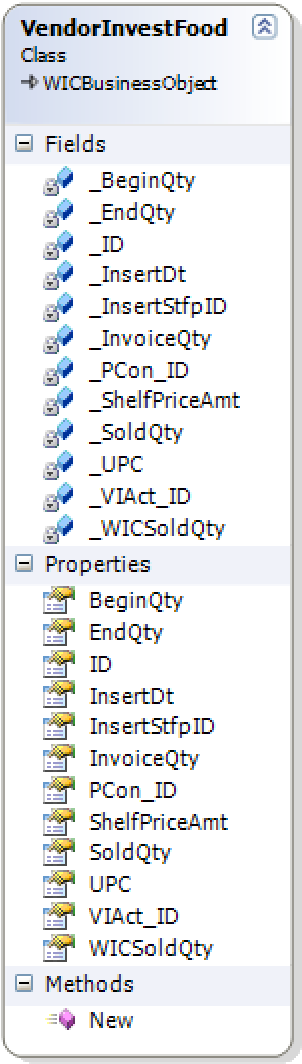


VendorInvestFI
Class
→ WICBusinessObject

Fields
- _ActualRedemptionAmt
- _Comments
- _ExpectedRedemptionAmt
- _ID
- _InsertDt
- _InsertStfpID
- _OutcomeCd
- _TransactionNr
- _VIAct_ID

Properties
- ActualRedemptionAmt
- Comments
- ExpectedRedemptionAmt
- ID
- InsertDt
- InsertStfpID
- OutcomeCd
- TransactionNr
- VIAct_ID

Methods
- New



VendorInvestFood
Class
→ WICBusinessObject

Fields

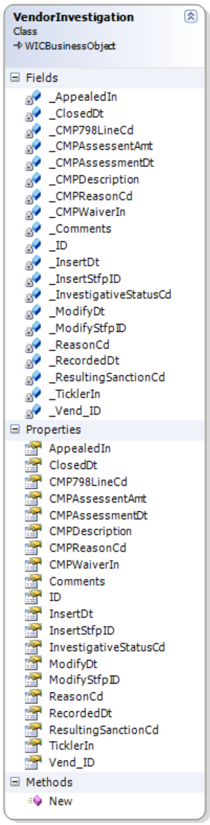
- _BeginQty
- _EndQty
- _ID
- _InsertDt
- _InsertStfpID
- _InvoiceQty
- _PCon_ID
- _ShelfPriceAmt
- _SoldQty
- _UPC
- _VIAct_ID
- _WICSoldQty

Properties

- BeginQty
- EndQty
- ID
- InsertDt
- InsertStfpID
- InvoiceQty
- PCon_ID
- ShelfPriceAmt
- SoldQty
- UPC
- VIAct_ID
- WICSoldQty

Methods

- New

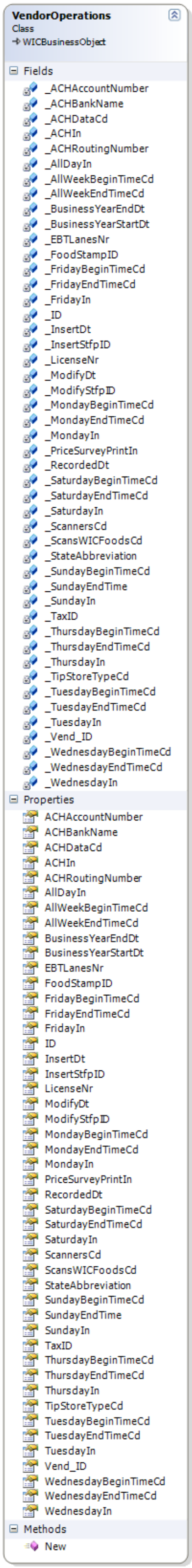


VendorInvestigation
Class
→ WICBusinessObject

Fields
- _AppealedIn
- _ClosedDt
- _CMP798LineCd
- _CMPAssessentAmt
- _CMPAssessmentDt
- _CMPDescription
- _CMPReasonCd
- _CMPWaiverIn
- _Comments
- _ID
- _InsertDt
- _InsertStfpID
- _InvestigativeStatusCd
- _ModifyDt
- _ModifyStfpID
- _ReasonCd
- _RecordedDt
- _ResultingSanctionCd
- _TicklerIn
- _Vend_ID

Properties
- AppealedIn
- ClosedDt
- CMP798LineCd
- CMPAssessentAmt
- CMPAssessmentDt
- CMPDescription
- CMPReasonCd
- CMPWaiverIn
- Comments
- ID
- InsertDt
- InsertStfpID
- InvestigativeStatusCd
- ModifyDt
- ModifyStfpID
- ReasonCd
- RecordedDt
- ResultingSanctionCd
- TicklerIn
- Vend_ID

Methods
- New

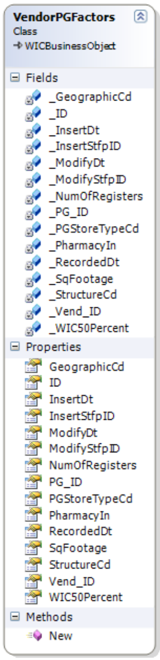


VendorOperations
Class
→ WICBusinessObject

Fields
- _ACHAccountNumber
- _ACHBankName
- _ACHDataCd
- _ACHIn
- _ACHRoutingNumber
- _AllDayIn
- _AllWeekBeginTimeCd
- _AllWeekEndTimeCd
- _BusinessYearEndDt
- _BusinessYearStartDt
- _EBTLanesNr
- _FoodStampID
- _FridayBeginTimeCd
- _FridayEndTimeCd
- _FridayIn
- _ID
- _InsertDt
- _InsertStfpID
- _LicenseNr
- _ModifyDt
- _ModifyStfpID
- _MondayBeginTimeCd
- _MondayEndTimeCd
- _MondayIn
- _PriceSurveyPrintIn
- _RecordedDt
- _SaturdayBeginTimeCd
- _SaturdayEndTimeCd
- _SaturdayIn
- _ScannersCd
- _ScansWICFoodsCd
- _StateAbbreviation
- _SundayBeginTimeCd
- _SundayEndTime
- _SundayIn
- _TaxID
- _ThursdayBeginTimeCd
- _ThursdayEndTimeCd
- _ThursdayIn
- _TipStoreTypeCd
- _TuesdayBeginTimeCd
- _TuesdayEndTimeCd
- _TuesdayIn
- _Vend_ID
- _WednesdayBeginTimeCd
- _WednesdayEndTimeCd
- _WednesdayIn

Properties
- ACHAccountNumber
- ACHBankName
- ACHDataCd
- ACHIn
- ACHRoutingNumber
- AllDayIn
- AllWeekBeginTimeCd
- AllWeekEndTimeCd
- BusinessYearEndDt
- BusinessYearStartDt
- EBTLanesNr
- FoodStampID
- FridayBeginTimeCd
- FridayEndTimeCd
- FridayIn
- ID
- InsertDt
- InsertStfpID
- LicenseNr
- ModifyDt
- ModifyStfpID
- MondayBeginTimeCd
- MondayEndTimeCd
- MondayIn
- PriceSurveyPrintIn
- RecordedDt
- SaturdayBeginTimeCd
- SaturdayEndTimeCd
- SaturdayIn
- ScannersCd
- ScansWICFoodsCd
- StateAbbreviation
- SundayBeginTimeCd
- SundayEndTime
- SundayIn
- TaxID
- ThursdayBeginTimeCd
- ThursdayEndTimeCd
- ThursdayIn
- TipStoreTypeCd
- TuesdayBeginTimeCd
- TuesdayEndTimeCd
- TuesdayIn
- Vend_ID
- WednesdayBeginTimeCd
- WednesdayEndTimeCd
- WednesdayIn

Methods
- New

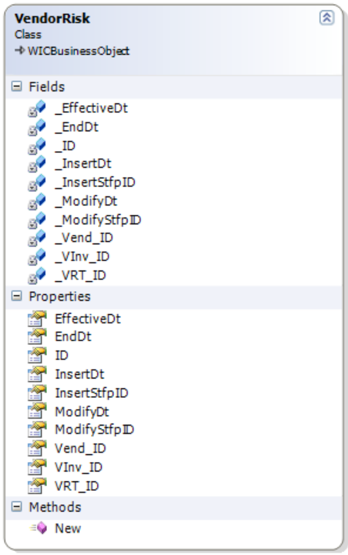


VendorPGFactors
Class
→ WICBusinessObject

Fields
- _GeographicCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _NumOfRegisters
- _PG_ID
- _PGStoreTypeCd
- _PharmacyIn
- _RecordedDt
- _SqFootage
- _StructureCd
- _Vend_ID
- _WIC50Percent

Properties
- GeographicCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- NumOfRegisters
- PG_ID
- PGStoreTypeCd
- PharmacyIn
- RecordedDt
- SqFootage
- StructureCd
- Vend_ID
- WIC50Percent

Methods
- New

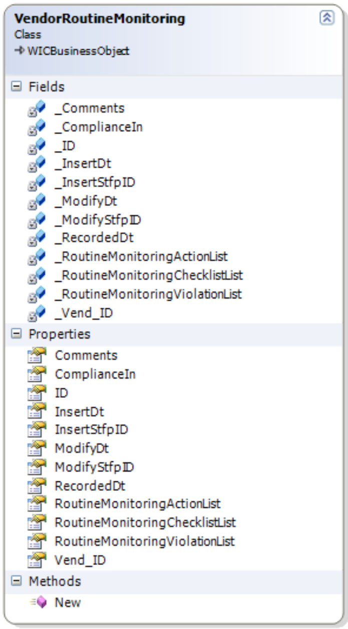


VendorRisk
Class
WICBusinessObject

Fields
- _EffectiveDt
- _EndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Vend_ID
- _VInv_ID
- _VRT_ID

Properties
- EffectiveDt
- EndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Vend_ID
- VInv_ID
- VRT_ID

Methods
- New

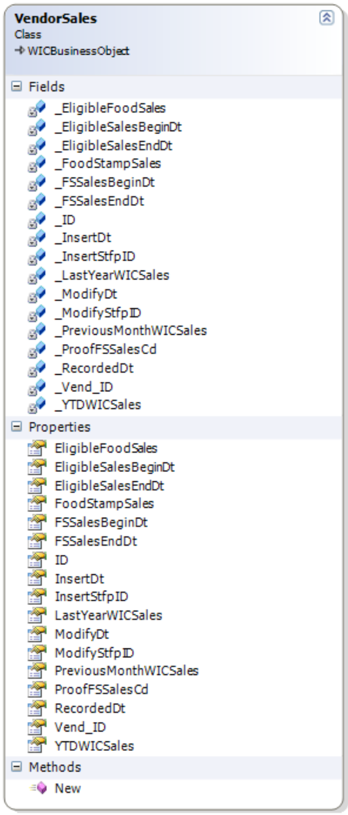


VendorRoutineMonitoring
Class
WICBusinessObject

Fields
- _Comments
- _ComplianceIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _RoutineMonitoringActionList
- _RoutineMonitoringChecklistList
- _RoutineMonitoringViolationList
- _Vend_ID

Properties
- Comments
- ComplianceIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- RoutineMonitoringActionList
- RoutineMonitoringChecklistList
- RoutineMonitoringViolationList
- Vend_ID

Methods
- New

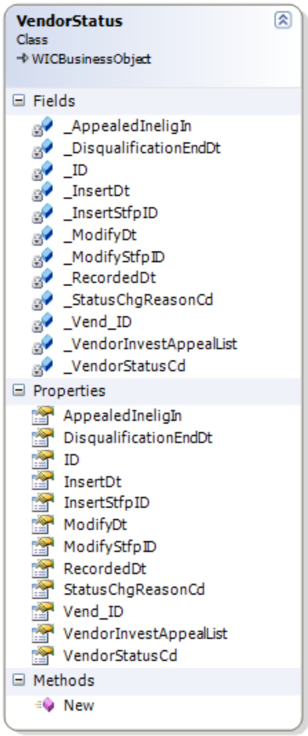


VendorSales
Class
WICBusinessObject

Fields
- _EligibleFoodSales
- _EligibleSalesBeginDt
- _EligibleSalesEndDt
- _FoodStampSales
- _FSSalesBeginDt
- _FSSalesEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _LastYearWICSales
- _ModifyDt
- _ModifyStfpID
- _PreviousMonthWICSales
- _ProofFSSalesCd
- _RecordedDt
- _Vend_ID
- _YTDWICSales

Properties
- EligibleFoodSales
- EligibleSalesBeginDt
- EligibleSalesEndDt
- FoodStampSales
- FSSalesBeginDt
- FSSalesEndDt
- ID
- InsertDt
- InsertStfpID
- LastYearWICSales
- ModifyDt
- ModifyStfpID
- PreviousMonthWICSales
- ProofFSSalesCd
- RecordedDt
- Vend_ID
- YTDWICSales

Methods
- New

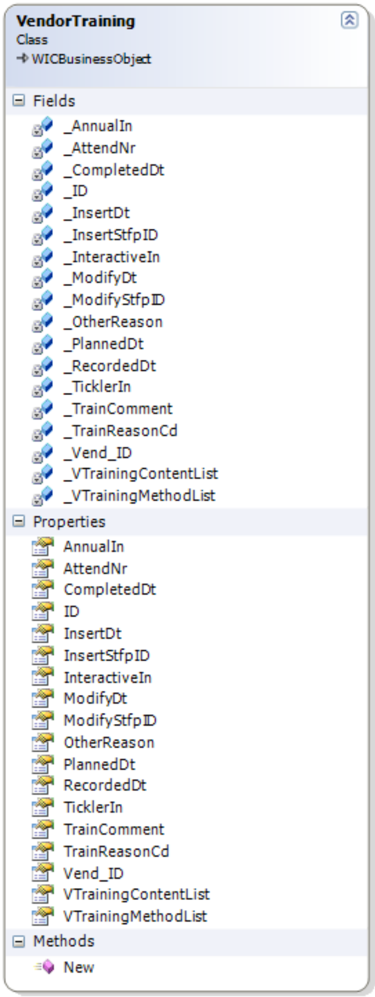


VendorStatus
Class
→ WICBusinessObject

Fields
- _AppealedIneligIn
- _DisqualificationEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _StatusChgReasonCd
- _Vend_ID
- _VendorInvestAppealList
- _VendorStatusCd

Properties
- AppealedIneligIn
- DisqualificationEndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- StatusChgReasonCd
- Vend_ID
- VendorInvestAppealList
- VendorStatusCd

Methods
- New

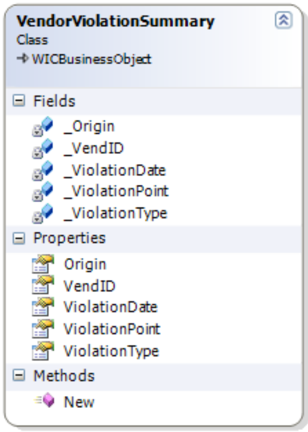


VendorTraining
Class
→ WICBusinessObject

Fields
- _AnnualIn
- _AttendNr
- _CompletedDt
- _ID
- _InsertDt
- _InsertStfpID
- _InteractiveIn
- _ModifyDt
- _ModifyStfpID
- _OtherReason
- _PlannedDt
- _RecordedDt
- _TicklerIn
- _TrainComment
- _TrainReasonCd
- _Vend_ID
- _VTrainingContentList
- _VTrainingMethodList

Properties
- AnnualIn
- AttendNr
- CompletedDt
- ID
- InsertDt
- InsertStfpID
- InteractiveIn
- ModifyDt
- ModifyStfpID
- OtherReason
- PlannedDt
- RecordedDt
- TicklerIn
- TrainComment
- TrainReasonCd
- Vend_ID
- VTrainingContentList
- VTrainingMethodList

Methods
- New

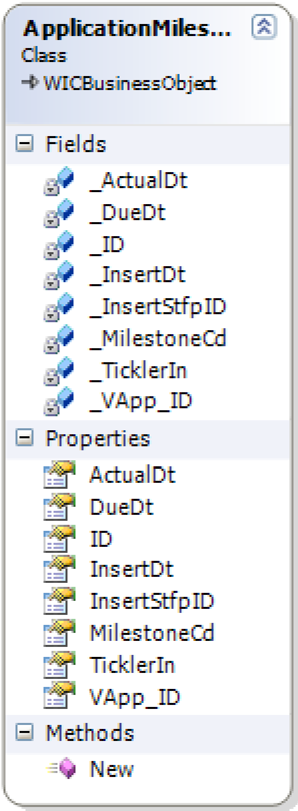


VendorViolationSummary
Class
→ WICBusinessObject

Fields
- _Origin
- _VendID
- _ViolationDate
- _ViolationPoint
- _ViolationType

Properties
- Origin
- VendID
- ViolationDate
- ViolationPoint
- ViolationType

Methods
- New

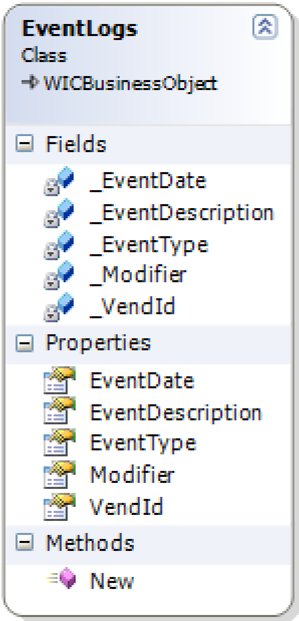


ApplicationMiles...
Class
→ WICBusinessObject

Fields
- _ActualDt
- _DueDt
- _ID
- _InsertDt
- _InsertStfpID
- _MilestoneCd
- _TicklerIn
- _VApp_ID

Properties
- ActualDt
- DueDt
- ID
- InsertDt
- InsertStfpID
- MilestoneCd
- TicklerIn
- VApp_ID

Methods
- New

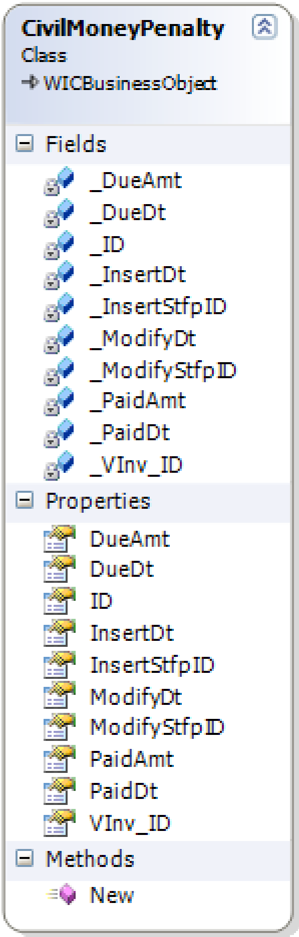


EventLogs
Class
→ WICBusinessObject

Fields
- _EventDate
- _EventDescription
- _EventType
- _Modifier
- _VendId

Properties
- EventDate
- EventDescription
- EventType
- Modifier
- VendId

Methods
- New

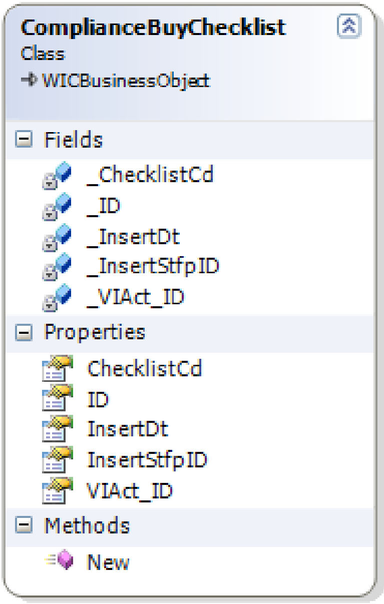


CivilMoneyPenalty
Class
→ WICBusinessObject

Fields
- _DueAmt
- _DueDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _PaidAmt
- _PaidDt
- _VInv_ID

Properties
- DueAmt
- DueDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- PaidAmt
- PaidDt
- VInv_ID

Methods
- New

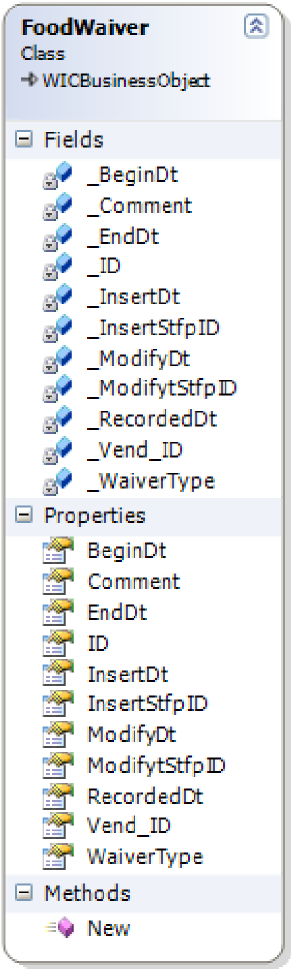


ComplianceBuyChecklist
Class
→ WICBusinessObject

Fields
- _ChecklistCd
- _ID
- _InsertDt
- _InsertStfpID
- _VIAct_ID

Properties
- ChecklistCd
- ID
- InsertDt
- InsertStfpID
- VIAct_ID

Methods
- New

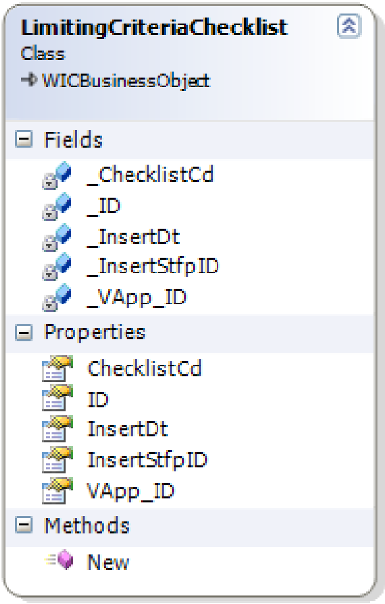


FoodWaiver
Class
→ WICBusinessObject

Fields
- _BeginDt
- _Comment
- _EndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifytStfpID
- _RecordedDt
- _Vend_ID
- _WaiverType

Properties
- BeginDt
- Comment
- EndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifytStfpID
- RecordedDt
- Vend_ID
- WaiverType

Methods
- New

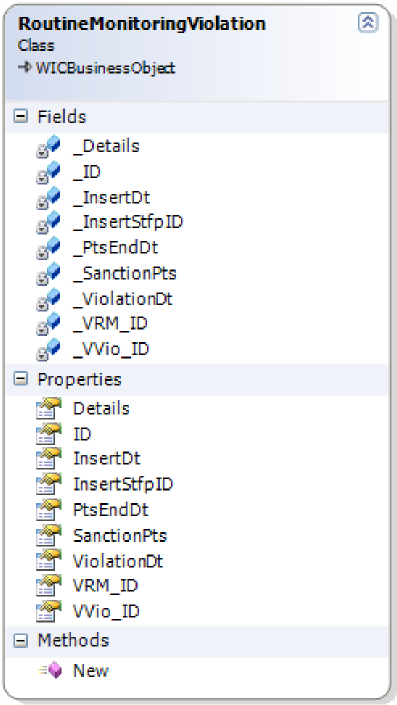


LimitingCriteriaChecklist
Class
→ WICBusinessObject

Fields
- _ChecklistCd
- _ID
- _InsertDt
- _InsertStfpID
- _VApp_ID

Properties
- ChecklistCd
- ID
- InsertDt
- InsertStfpID
- VApp_ID

Methods
- New

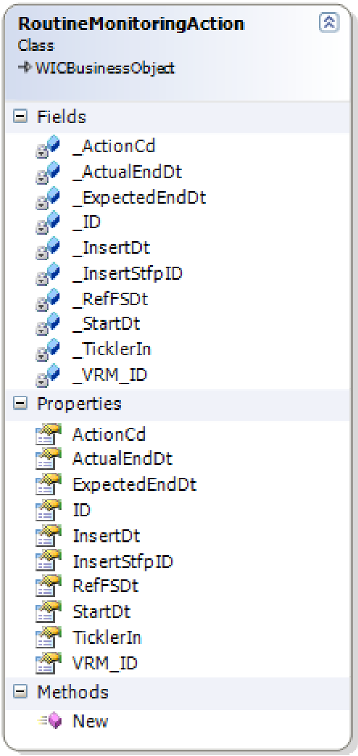


RoutineMonitoringViolation
Class
WICBusinessObject

Fields
- _Details
- _ID
- _InsertDt
- _InsertStfpID
- _PtsEndDt
- _SanctionPts
- _ViolationDt
- _VRM_ID
- _VVio_ID

Properties
- Details
- ID
- InsertDt
- InsertStfpID
- PtsEndDt
- SanctionPts
- ViolationDt
- VRM_ID
- VVio_ID

Methods
- New

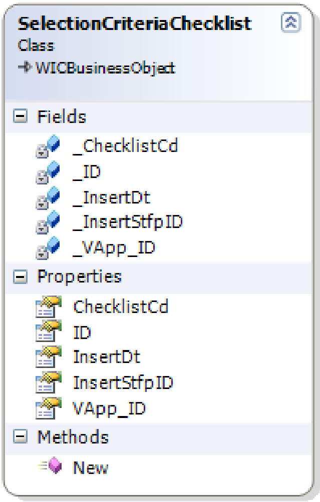


RoutineMonitoringAction
Class
WICBusinessObject

Fields
- _ActionCd
- _ActualEndDt
- _ExpectedEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _RefFSDt
- _StartDt
- _TicklerIn
- _VRM_ID

Properties
- ActionCd
- ActualEndDt
- ExpectedEndDt
- ID
- InsertDt
- InsertStfpID
- RefFSDt
- StartDt
- TicklerIn
- VRM_ID

Methods
- New

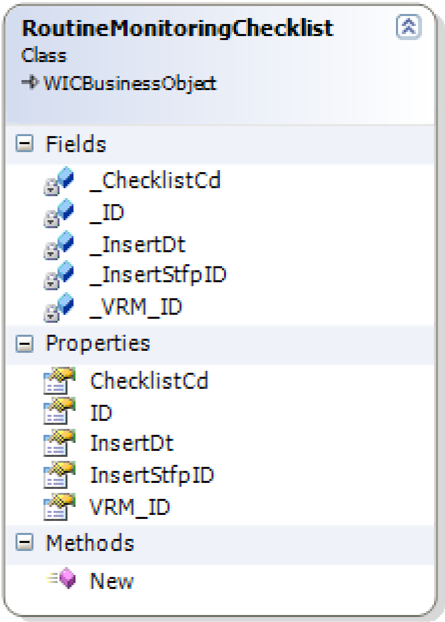


SelectionCriteriaChecklist
Class
WICBusinessObject

Fields
- _ChecklistCd
- _ID
- _InsertDt
- _InsertStfpID
- _VApp_ID

Properties
- ChecklistCd
- ID
- InsertDt
- InsertStfpID
- VApp_ID

Methods
- New

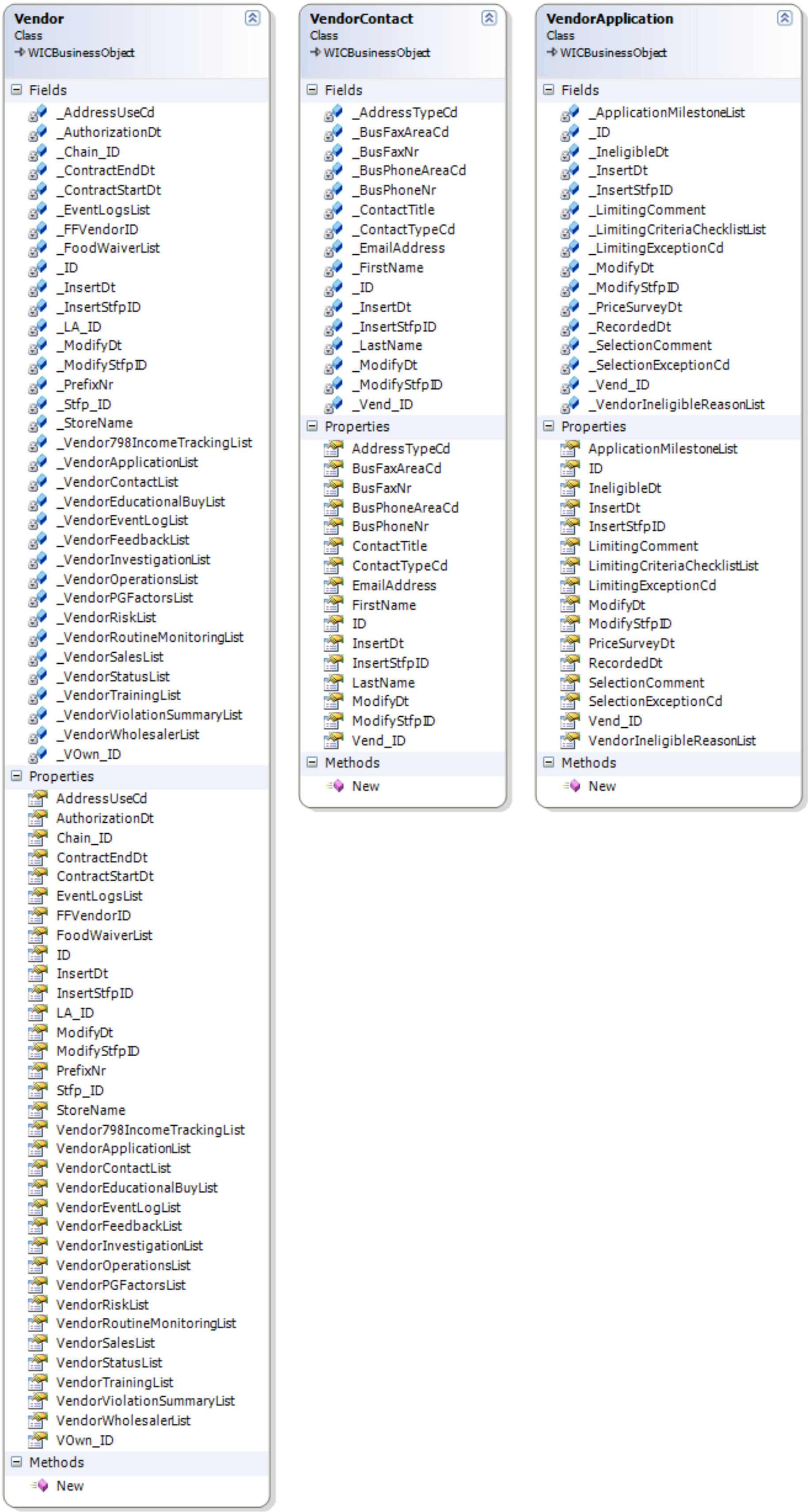


RoutineMonitoringChecklist
Class
WICBusinessObject

Fields
- _ChecklistCd
- _ID
- _InsertDt
- _InsertStfpID
- _VRM_ID

Properties
- ChecklistCd
- ID
- InsertDt
- InsertStfpID
- VRM_ID

Methods
- New



Vendor
Class
→ WICBusinessObject

Fields
- _AddressUseCd
- _AuthorizationDt
- _Chain_ID
- _ContractEndDt
- _ContractStartDt
- _EventLogsList
- _FFVendorID
- _FoodWaiverList
- _ID
- _InsertDt
- _InsertStfpID
- _LA_ID
- _ModifyDt
- _ModifyStfpID
- _PrefixNr
- _Stfp_ID
- _StoreName
- _Vendor798IncomeTrackingList
- _VendorApplicationList
- _VendorContactList
- _VendorEducationalBuyList
- _VendorEventLogList
- _VendorFeedbackList
- _VendorInvestigationList
- _VendorOperationsList
- _VendorPGFactorsList
- _VendorRiskList
- _VendorRoutineMonitoringList
- _VendorSalesList
- _VendorStatusList
- _VendorTrainingList
- _VendorViolationSummaryList
- _VendorWholesalerList
- _VOwn_ID

Properties
- AddressUseCd
- AuthorizationDt
- Chain_ID
- ContractEndDt
- ContractStartDt
- EventLogsList
- FFVendorID
- FoodWaiverList
- ID
- InsertDt
- InsertStfpID
- LA_ID
- ModifyDt
- ModifyStfpID
- PrefixNr
- Stfp_ID
- StoreName
- Vendor798IncomeTrackingList
- VendorApplicationList
- VendorContactList
- VendorEducationalBuyList
- VendorEventLogList
- VendorFeedbackList
- VendorInvestigationList
- VendorOperationsList
- VendorPGFactorsList
- VendorRiskList
- VendorRoutineMonitoringList
- VendorSalesList
- VendorStatusList
- VendorTrainingList
- VendorViolationSummaryList
- VendorWholesalerList
- VOwn_ID

Methods
- New

VendorContact
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _BusFaxAreaCd
- _BusFaxNr
- _BusPhoneAreaCd
- _BusPhoneNr
- _ContactTitle
- _ContactTypeCd
- _EmailAddress
- _FirstName
- _ID
- _InsertDt
- _InsertStfpID
- _LastName
- _ModifyDt
- _ModifyStfpID
- _Vend_ID

Properties
- AddressTypeCd
- BusFaxAreaCd
- BusFaxNr
- BusPhoneAreaCd
- BusPhoneNr
- ContactTitle
- ContactTypeCd
- EmailAddress
- FirstName
- ID
- InsertDt
- InsertStfpID
- LastName
- ModifyDt
- ModifyStfpID
- Vend_ID

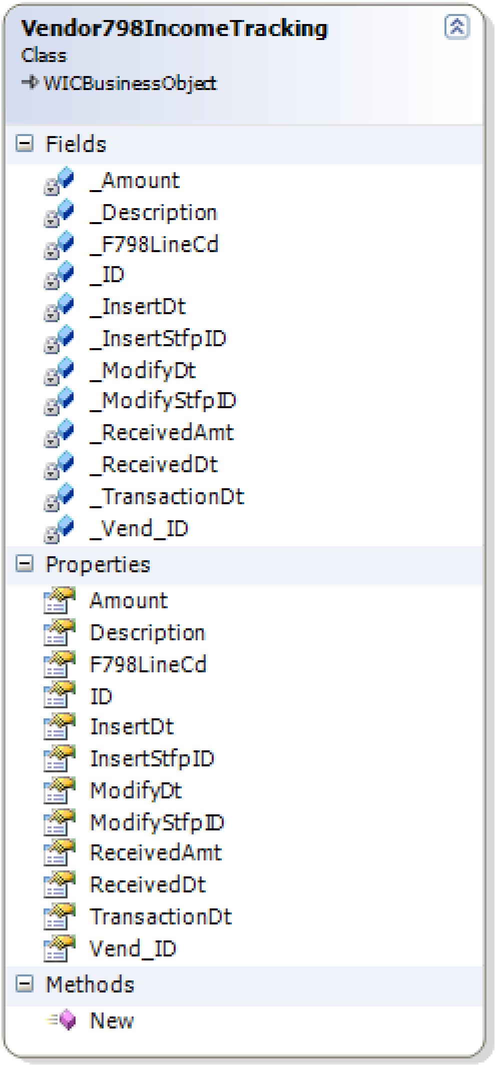
Methods
- New

VendorApplication
Class
→ WICBusinessObject

Fields
- _ApplicationMilestoneList
- _ID
- _IneligibleDt
- _InsertDt
- _InsertStfpID
- _LimitingComment
- _LimitingCriteriaChecklistList
- _LimitingExceptionCd
- _ModifyDt
- _ModifyStfpID
- _PriceSurveyDt
- _RecordedDt
- _SelectionComment
- _SelectionExceptionCd
- _Vend_ID
- _VendorIneligibleReasonList

Properties
- ApplicationMilestoneList
- ID
- IneligibleDt
- InsertDt
- InsertStfpID
- LimitingComment
- LimitingCriteriaChecklistList
- LimitingExceptionCd
- ModifyDt
- ModifyStfpID
- PriceSurveyDt
- RecordedDt
- SelectionComment
- SelectionExceptionCd
- Vend_ID
- VendorIneligibleReasonList

Methods
- New

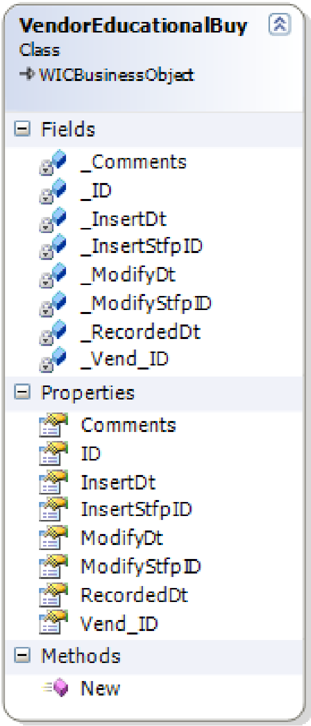


Vendor798IncomeTracking
Class
→ WICBusinessObject

Fields
- _Amount
- _Description
- _F798LineCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _ReceivedAmt
- _ReceivedDt
- _TransactionDt
- _Vend_ID

Properties
- Amount
- Description
- F798LineCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- ReceivedAmt
- ReceivedDt
- TransactionDt
- Vend_ID

Methods
- New

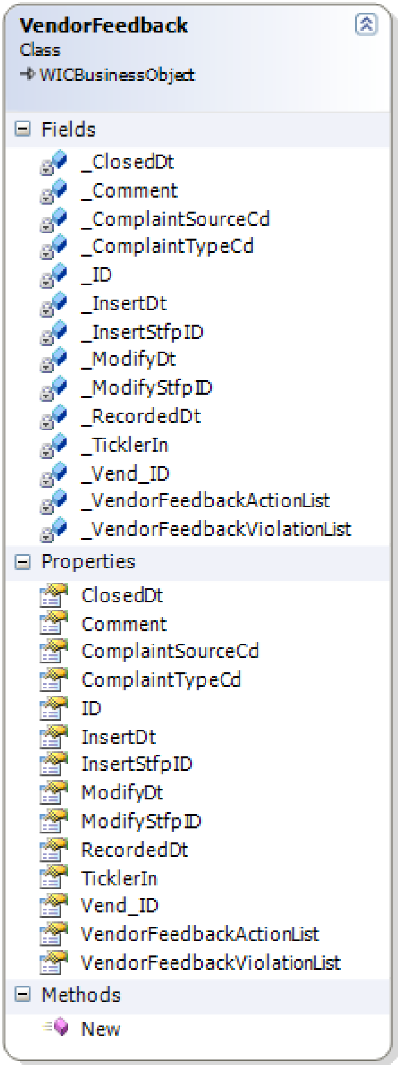


VendorEducationalBuy
Class
→ WICBusinessObject

Fields
- _Comments
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _Vend_ID

Properties
- Comments
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- Vend_ID

Methods
- New

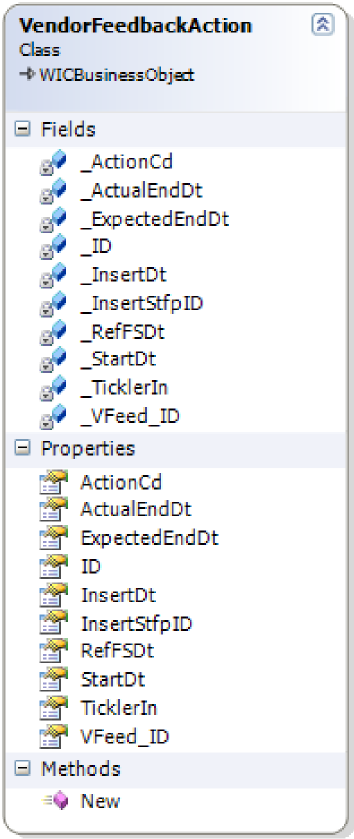


VendorFeedback
Class
→ WICBusinessObject

Fields
- _ClosedDt
- _Comment
- _ComplaintSourceCd
- _ComplaintTypeCd
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _TicklerIn
- _Vend_ID
- _VendorFeedbackActionList
- _VendorFeedbackViolationList

Properties
- ClosedDt
- Comment
- ComplaintSourceCd
- ComplaintTypeCd
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- TicklerIn
- Vend_ID
- VendorFeedbackActionList
- VendorFeedbackViolationList

Methods
- New

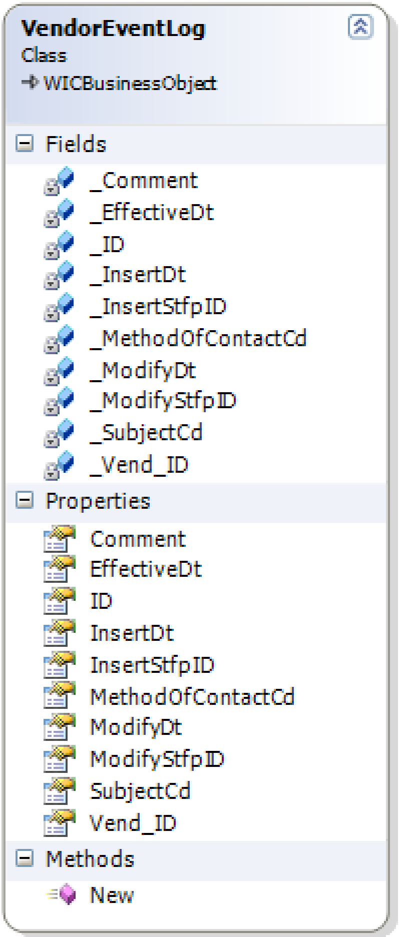


VendorFeedbackAction
Class
→ WICBusinessObject

Fields
- _ActionCd
- _ActualEndDt
- _ExpectedEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _RefFSDt
- _StartDt
- _TicklerIn
- _VFeed_ID

Properties
- ActionCd
- ActualEndDt
- ExpectedEndDt
- ID
- InsertDt
- InsertStfpID
- RefFSDt
- StartDt
- TicklerIn
- VFeed_ID

Methods
- New

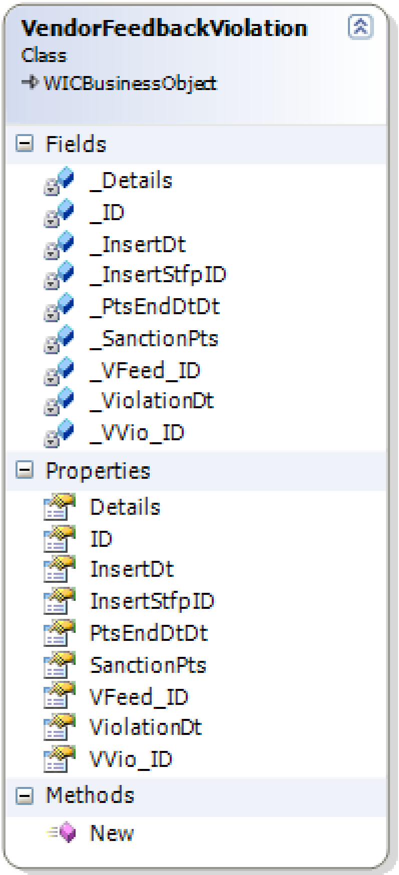


VendorEventLog
Class
WICBusinessObject

Fields
- _Comment
- _EffectiveDt
- _ID
- _InsertDt
- _InsertStfpID
- _MethodOfContactCd
- _ModifyDt
- _ModifyStfpID
- _SubjectCd
- _Vend_ID

Properties
- Comment
- EffectiveDt
- ID
- InsertDt
- InsertStfpID
- MethodOfContactCd
- ModifyDt
- ModifyStfpID
- SubjectCd
- Vend_ID

Methods
- New

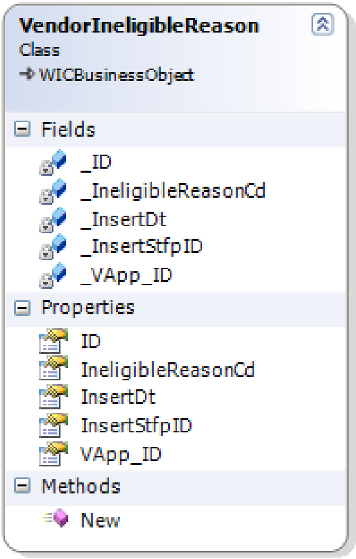


VendorFeedbackViolation
Class
WICBusinessObject

Fields
- _Details
- _ID
- _InsertDt
- _InsertStfpID
- _PtsEndDtDt
- _SanctionPts
- _VFeed_ID
- _ViolationDt
- _VVio_ID

Properties
- Details
- ID
- InsertDt
- InsertStfpID
- PtsEndDtDt
- SanctionPts
- VFeed_ID
- ViolationDt
- VVio_ID

Methods
- New

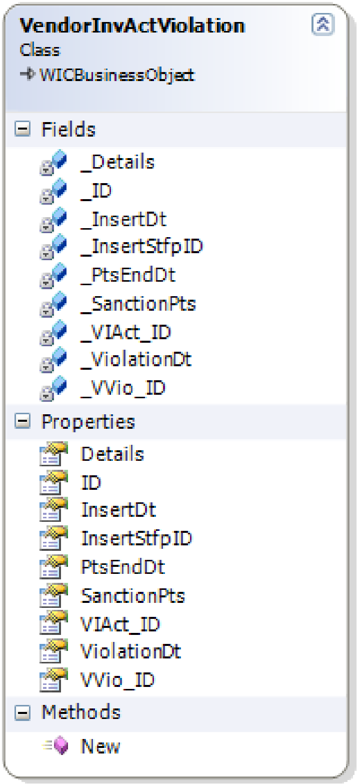


VendorIneligibleReason
Class
→ WICBusinessObject

Fields
- _ID
- _IneligibleReasonCd
- _InsertDt
- _InsertStfpID
- _VApp_ID

Properties
- ID
- IneligibleReasonCd
- InsertDt
- InsertStfpID
- VApp_ID

Methods
- New

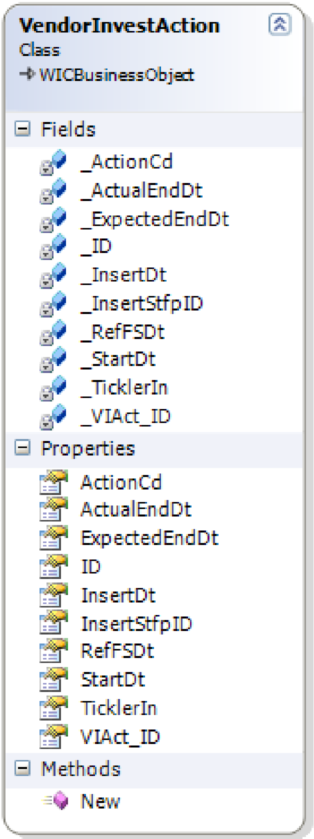


VendorInvActViolation
Class
→ WICBusinessObject

Fields
- _Details
- _ID
- _InsertDt
- _InsertStfpID
- _PtsEndDt
- _SanctionPts
- _VIAct_ID
- _ViolationDt
- _VVio_ID

Properties
- Details
- ID
- InsertDt
- InsertStfpID
- PtsEndDt
- SanctionPts
- VIAct_ID
- ViolationDt
- VVio_ID

Methods
- New

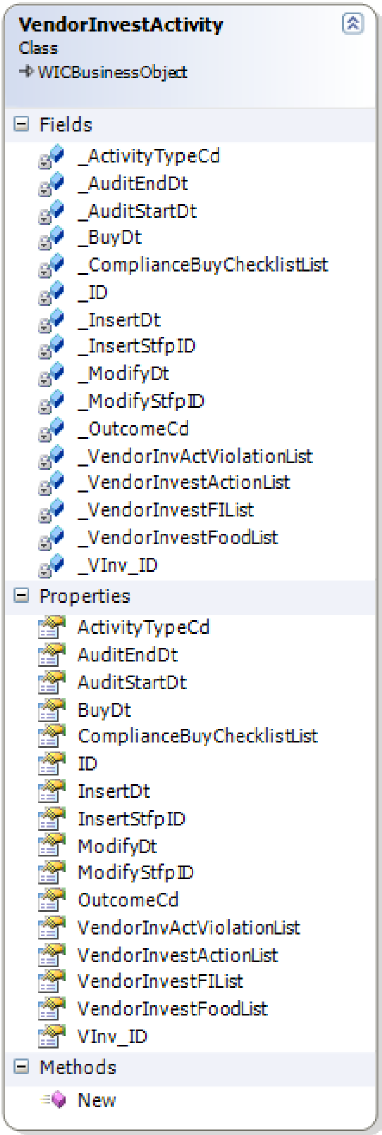


VendorInvestAction
Class
→ WICBusinessObject

Fields
- _ActionCd
- _ActualEndDt
- _ExpectedEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _RefFSDt
- _StartDt
- _TicklerIn
- _VIAct_ID

Properties
- ActionCd
- ActualEndDt
- ExpectedEndDt
- ID
- InsertDt
- InsertStfpID
- RefFSDt
- StartDt
- TicklerIn
- VIAct_ID

Methods
- New

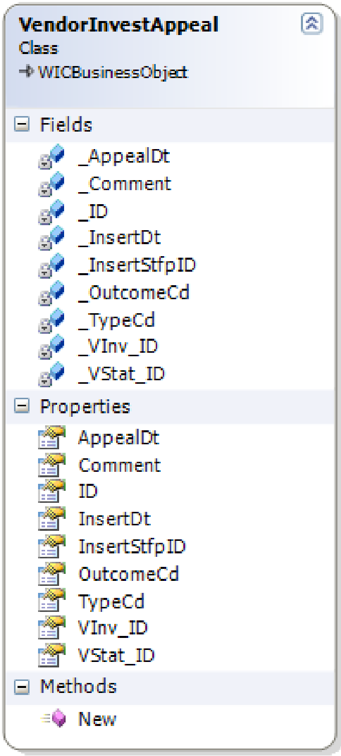


VendorInvestActivity
Class
→ WICBusinessObject

Fields
- _ActivityTypeCd
- _AuditEndDt
- _AuditStartDt
- _BuyDt
- _ComplianceBuyChecklistList
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _OutcomeCd
- _VendorInvActViolationList
- _VendorInvestActionList
- _VendorInvestFIList
- _VendorInvestFoodList
- _VInv_ID

Properties
- ActivityTypeCd
- AuditEndDt
- AuditStartDt
- BuyDt
- ComplianceBuyChecklistList
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- OutcomeCd
- VendorInvActViolationList
- VendorInvestActionList
- VendorInvestFIList
- VendorInvestFoodList
- VInv_ID

Methods
- New

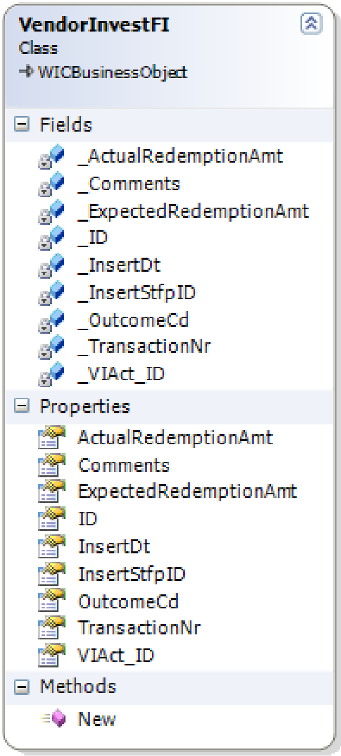


VendorInvestAppeal
Class
→ WICBusinessObject

Fields
- _AppealDt
- _Comment
- _ID
- _InsertDt
- _InsertStfpID
- _OutcomeCd
- _TypeCd
- _VInv_ID
- _VStat_ID

Properties
- AppealDt
- Comment
- ID
- InsertDt
- InsertStfpID
- OutcomeCd
- TypeCd
- VInv_ID
- VStat_ID

Methods
- New

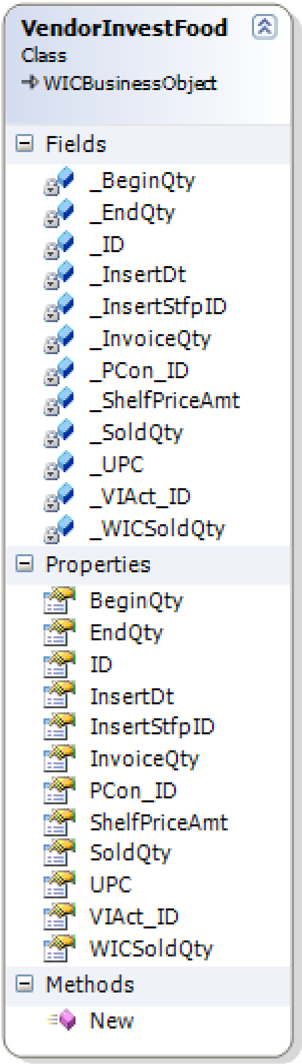


VendorInvestFI
Class
→ WICBusinessObject

Fields
- _ActualRedemptionAmt
- _Comments
- _ExpectedRedemptionAmt
- _ID
- _InsertDt
- _InsertStfpID
- _OutcomeCd
- _TransactionNr
- _VIAct_ID

Properties
- ActualRedemptionAmt
- Comments
- ExpectedRedemptionAmt
- ID
- InsertDt
- InsertStfpID
- OutcomeCd
- TransactionNr
- VIAct_ID

Methods
- New



VendorInvestFood
Class
➔ WICBusinessObject

⊟ Fields

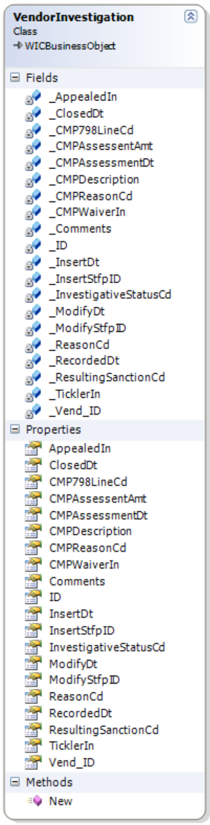
- _BeginQty
- _EndQty
- _ID
- _InsertDt
- _InsertStfpID
- _InvoiceQty
- _PCon_ID
- _ShelfPriceAmt
- _SoldQty
- _UPC
- _VIAct_ID
- _WICSoldQty

⊟ Properties

- BeginQty
- EndQty
- ID
- InsertDt
- InsertStfpID
- InvoiceQty
- PCon_ID
- ShelfPriceAmt
- SoldQty
- UPC
- VIAct_ID
- WICSoldQty

⊟ Methods

- New

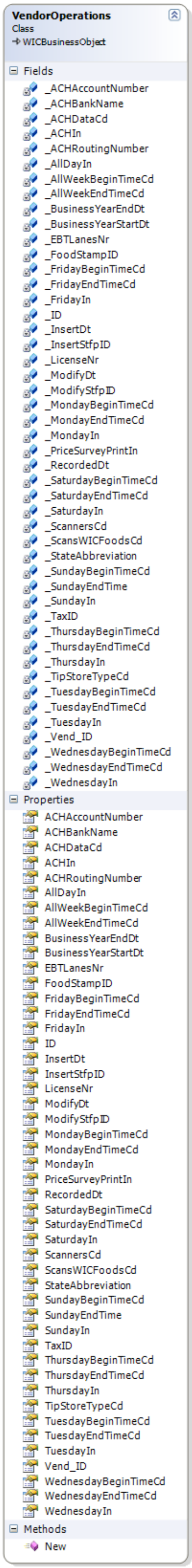


VendorInvestigation
Class
→ WICBusinessObject

⊟ Fields
_AppealedIn
_ClosedDt
_CMP798LineCd
_CMPAssessentAmt
_CMPAssessmentDt
_CMPDescription
_CMPReasonCd
_CMPWaiverIn
_Comments
_ID
_InsertDt
_InsertStfpID
_InvestigativeStatusCd
_ModifyDt
_ModifyStfpID
_ReasonCd
_RecordedDt
_ResultingSanctionCd
_TicklerIn
_Vend_ID

⊟ Properties
AppealedIn
ClosedDt
CMP798LineCd
CMPAssessentAmt
CMPAssessmentDt
CMPDescription
CMPReasonCd
CMPWaiverIn
Comments
ID
InsertDt
InsertStfpID
InvestigativeStatusCd
ModifyDt
ModifyStfpID
ReasonCd
RecordedDt
ResultingSanctionCd
TicklerIn
Vend_ID

⊟ Methods
New

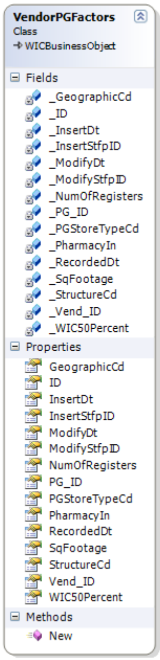


VendorOperations
Class
→ WICBusinessObject

⊟ Fields
_ACHAccountNumber
_ACHBankName
_ACHDataCd
_ACHIn
_ACHRoutingNumber
_AllDayIn
_AllWeekBeginTimeCd
_AllWeekEndTimeCd
_BusinessYearEndDt
_BusinessYearStartDt
_EBTLanesNr
_FoodStampID
_FridayBeginTimeCd
_FridayEndTimeCd
_FridayIn
_ID
_InsertDt
_InsertStfpID
_LicenseNr
_ModifyDt
_ModifyStfpID
_MondayBeginTimeCd
_MondayEndTimeCd
_MondayIn
_PriceSurveyPrintIn
_RecordedDt
_SaturdayBeginTimeCd
_SaturdayEndTimeCd
_SaturdayIn
_ScannersCd
_ScansWICFoodsCd
_StateAbbreviation
_SundayBeginTimeCd
_SundayEndTime
_SundayIn
_TaxID
_ThursdayBeginTimeCd
_ThursdayEndTimeCd
_ThursdayIn
_TipStoreTypeCd
_TuesdayBeginTimeCd
_TuesdayEndTimeCd
_TuesdayIn
_Vend_ID
_WednesdayBeginTimeCd
_WednesdayEndTimeCd
_WednesdayIn

⊟ Properties
ACHAccountNumber
ACHBankName
ACHDataCd
ACHIn
ACHRoutingNumber
AllDayIn
AllWeekBeginTimeCd
AllWeekEndTimeCd
BusinessYearEndDt
BusinessYearStartDt
EBTLanesNr
FoodStampID
FridayBeginTimeCd
FridayEndTimeCd
FridayIn
ID
InsertDt
InsertStfpID
LicenseNr
ModifyDt
ModifyStfpID
MondayBeginTimeCd
MondayEndTimeCd
MondayIn
PriceSurveyPrintIn
RecordedDt
SaturdayBeginTimeCd
SaturdayEndTimeCd
SaturdayIn
ScannersCd
ScansWICFoodsCd
StateAbbreviation
SundayBeginTimeCd
SundayEndTime
SundayIn
TaxID
ThursdayBeginTimeCd
ThursdayEndTimeCd
ThursdayIn
TipStoreTypeCd
TuesdayBeginTimeCd
TuesdayEndTimeCd
TuesdayIn
Vend_ID
WednesdayBeginTimeCd
WednesdayEndTimeCd
WednesdayIn

⊟ Methods
New

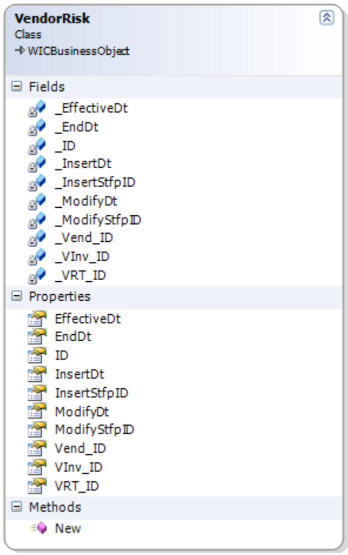


VendorPGFactors
Class
→ WICBusinessObject

⊟ Fields
_GeographicCd
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_NumOfRegisters
_PG_ID
_PGStoreTypeCd
_PharmacyIn
_RecordedDt
_SqFootage
_StructureCd
_Vend_ID
_WIC50Percent

⊟ Properties
GeographicCd
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
NumOfRegisters
PG_ID
PGStoreTypeCd
PharmacyIn
RecordedDt
SqFootage
StructureCd
Vend_ID
WIC50Percent

⊟ Methods
New

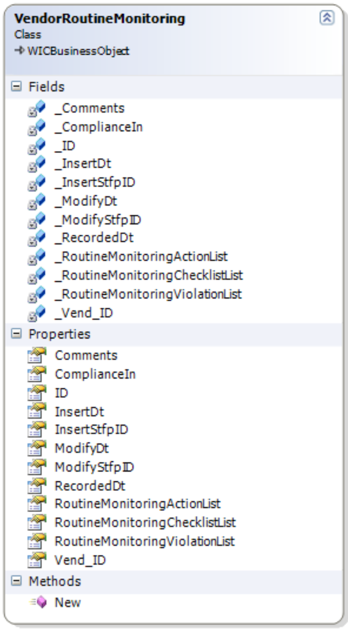


VendorRisk
Class
→ WICBusinessObject

Fields
- _EffectiveDt
- _EndDt
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _Vend_ID
- _VInv_ID
- _VRT_ID

Properties
- EffectiveDt
- EndDt
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- Vend_ID
- VInv_ID
- VRT_ID

Methods
- New

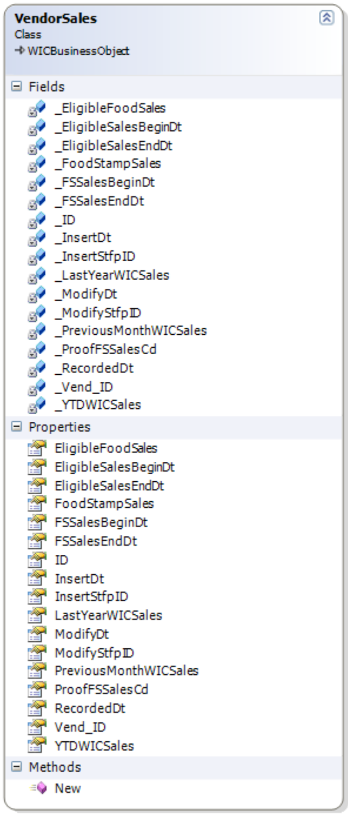


VendorRoutineMonitoring
Class
→ WICBusinessObject

Fields
- _Comments
- _ComplianceIn
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _RecordedDt
- _RoutineMonitoringActionList
- _RoutineMonitoringChecklistList
- _RoutineMonitoringViolationList
- _Vend_ID

Properties
- Comments
- ComplianceIn
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- RecordedDt
- RoutineMonitoringActionList
- RoutineMonitoringChecklistList
- RoutineMonitoringViolationList
- Vend_ID

Methods
- New

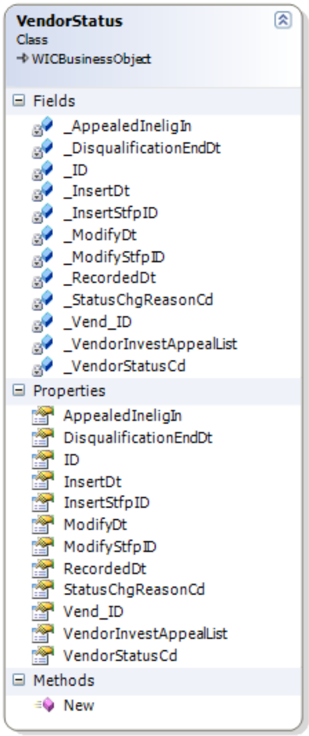


VendorSales
Class
→ WICBusinessObject

Fields
- _EligibleFoodSales
- _EligibleSalesBeginDt
- _EligibleSalesEndDt
- _FoodStampSales
- _FSSalesBeginDt
- _FSSalesEndDt
- _ID
- _InsertDt
- _InsertStfpID
- _LastYearWICSales
- _ModifyDt
- _ModifyStfpID
- _PreviousMonthWICSales
- _ProofFSSalesCd
- _RecordedDt
- _Vend_ID
- _YTDWICSales

Properties
- EligibleFoodSales
- EligibleSalesBeginDt
- EligibleSalesEndDt
- FoodStampSales
- FSSalesBeginDt
- FSSalesEndDt
- ID
- InsertDt
- InsertStfpID
- LastYearWICSales
- ModifyDt
- ModifyStfpID
- PreviousMonthWICSales
- ProofFSSalesCd
- RecordedDt
- Vend_ID
- YTDWICSales

Methods
- New

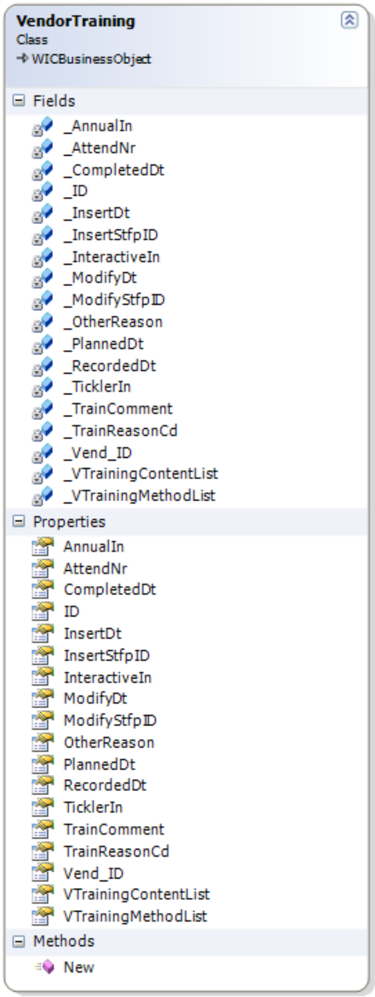


VendorStatus
Class
WICBusinessObject

Fields
_AppealedIneligIn
_DisqualificationEndDt
_ID
_InsertDt
_InsertStfpID
_ModifyDt
_ModifyStfpID
_RecordedDt
_StatusChgReasonCd
_Vend_ID
_VendorInvestAppealList
_VendorStatusCd

Properties
AppealedIneligIn
DisqualificationEndDt
ID
InsertDt
InsertStfpID
ModifyDt
ModifyStfpID
RecordedDt
StatusChgReasonCd
Vend_ID
VendorInvestAppealList
VendorStatusCd

Methods
New

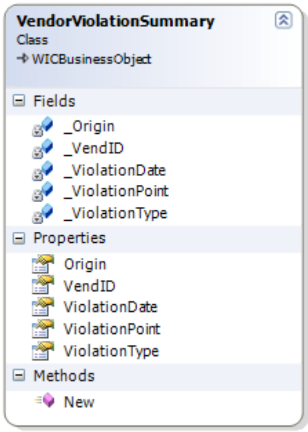


VendorTraining
Class
WICBusinessObject

Fields
_AnnualIn
_AttendNr
_CompletedDt
_ID
_InsertDt
_InsertStfpID
_InteractiveIn
_ModifyDt
_ModifyStfpID
_OtherReason
_PlannedDt
_RecordedDt
_TicklerIn
_TrainComment
_TrainReasonCd
_Vend_ID
_VTrainingContentList
_VTrainingMethodList

Properties
AnnualIn
AttendNr
CompletedDt
ID
InsertDt
InsertStfpID
InteractiveIn
ModifyDt
ModifyStfpID
OtherReason
PlannedDt
RecordedDt
TicklerIn
TrainComment
TrainReasonCd
Vend_ID
VTrainingContentList
VTrainingMethodList

Methods
New

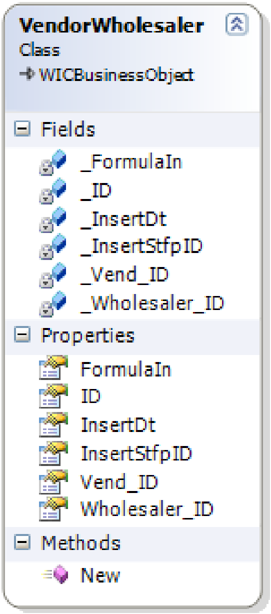


VendorViolationSummary
Class
WICBusinessObject

Fields
_Origin
_VendID
_ViolationDate
_ViolationPoint
_ViolationType

Properties
Origin
VendID
ViolationDate
ViolationPoint
ViolationType

Methods
New

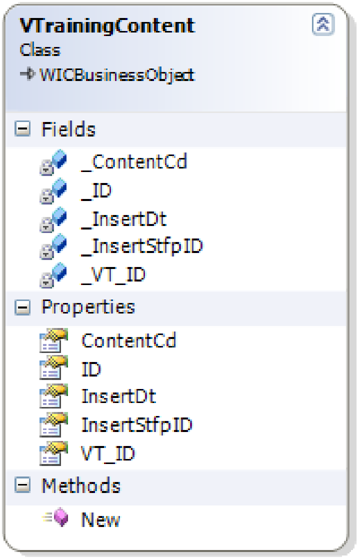


VendorWholesaler
Class
→ WICBusinessObject

⊟ Fields
- _FormulaIn
- _ID
- _InsertDt
- _InsertStfpID
- _Vend_ID
- _Wholesaler_ID

⊟ Properties
- FormulaIn
- ID
- InsertDt
- InsertStfpID
- Vend_ID
- Wholesaler_ID

⊟ Methods
- New

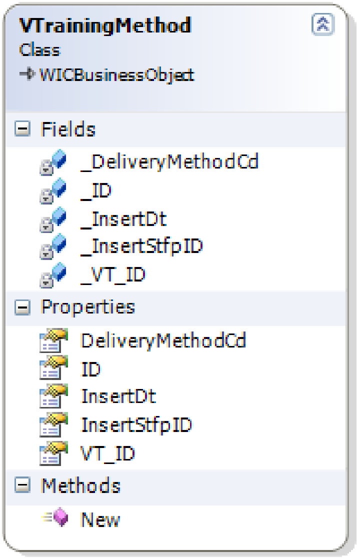


VTrainingContent
Class
→ WICBusinessObject

⊟ Fields
- _ContentCd
- _ID
- _InsertDt
- _InsertStfpID
- _VT_ID

⊟ Properties
- ContentCd
- ID
- InsertDt
- InsertStfpID
- VT_ID

⊟ Methods
- New

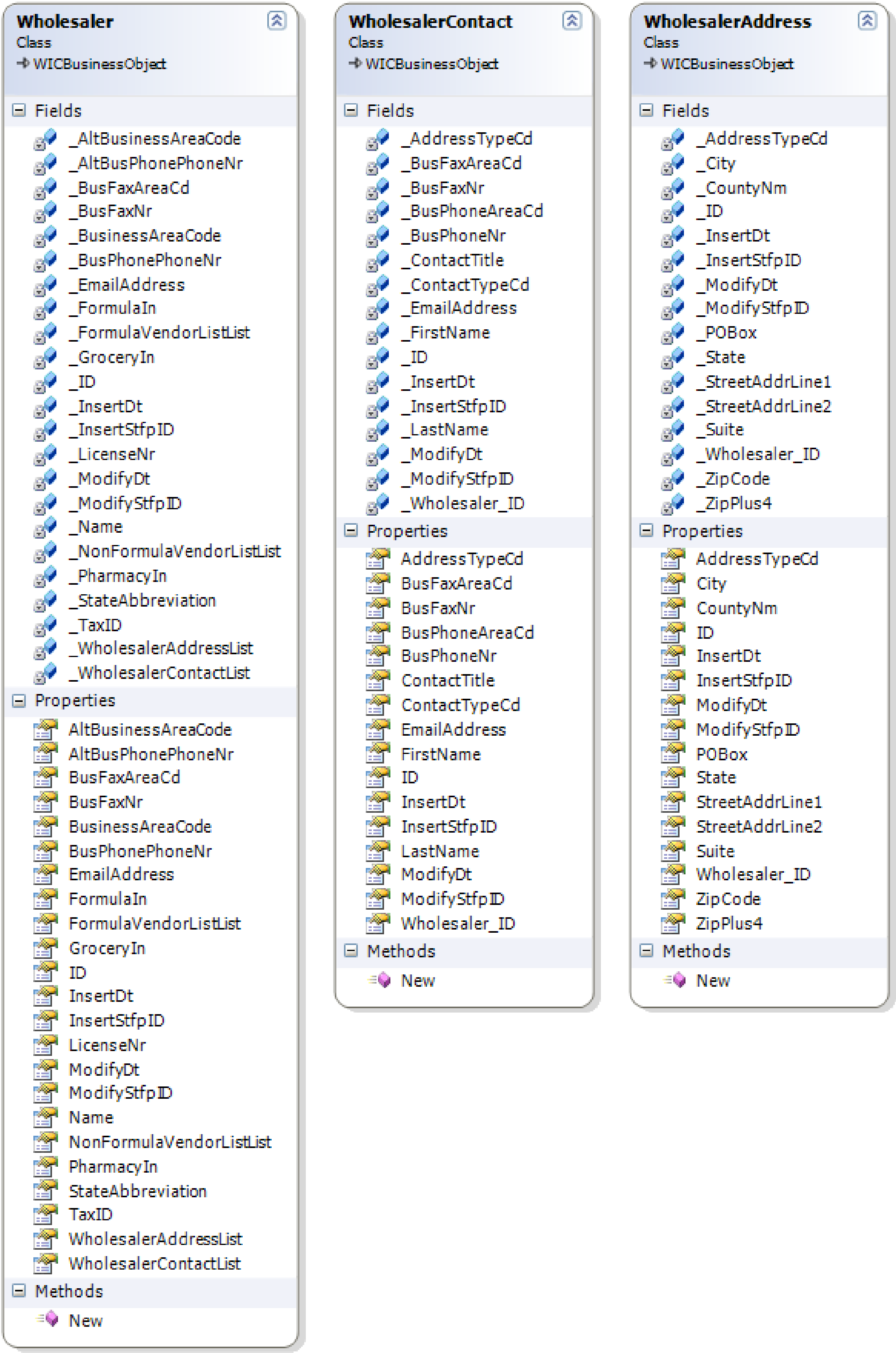


VTrainingMethod
Class
→ WICBusinessObject

⊟ Fields
- _DeliveryMethodCd
- _ID
- _InsertDt
- _InsertStfpID
- _VT_ID

⊟ Properties
- DeliveryMethodCd
- ID
- InsertDt
- InsertStfpID
- VT_ID

⊟ Methods
- New



Wholesaler
Class
→ WICBusinessObject

Fields
- _AltBusinessAreaCode
- _AltBusPhonePhoneNr
- _BusFaxAreaCd
- _BusFaxNr
- _BusinessAreaCode
- _BusPhonePhoneNr
- _EmailAddress
- _FormulaIn
- _FormulaVendorListList
- _GroceryIn
- _ID
- _InsertDt
- _InsertStfpID
- _LicenseNr
- _ModifyDt
- _ModifyStfpID
- _Name
- _NonFormulaVendorListList
- _PharmacyIn
- _StateAbbreviation
- _TaxID
- _WholesalerAddressList
- _WholesalerContactList

Properties
- AltBusinessAreaCode
- AltBusPhonePhoneNr
- BusFaxAreaCd
- BusFaxNr
- BusinessAreaCode
- BusPhonePhoneNr
- EmailAddress
- FormulaIn
- FormulaVendorListList
- GroceryIn
- ID
- InsertDt
- InsertStfpID
- LicenseNr
- ModifyDt
- ModifyStfpID
- Name
- NonFormulaVendorListList
- PharmacyIn
- StateAbbreviation
- TaxID
- WholesalerAddressList
- WholesalerContactList

Methods
- New

WholesalerContact
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _BusFaxAreaCd
- _BusFaxNr
- _BusPhoneAreaCd
- _BusPhoneNr
- _ContactTitle
- _ContactTypeCd
- _EmailAddress
- _FirstName
- _ID
- _InsertDt
- _InsertStfpID
- _LastName
- _ModifyDt
- _ModifyStfpID
- _Wholesaler_ID

Properties
- AddressTypeCd
- BusFaxAreaCd
- BusFaxNr
- BusPhoneAreaCd
- BusPhoneNr
- ContactTitle
- ContactTypeCd
- EmailAddress
- FirstName
- ID
- InsertDt
- InsertStfpID
- LastName
- ModifyDt
- ModifyStfpID
- Wholesaler_ID

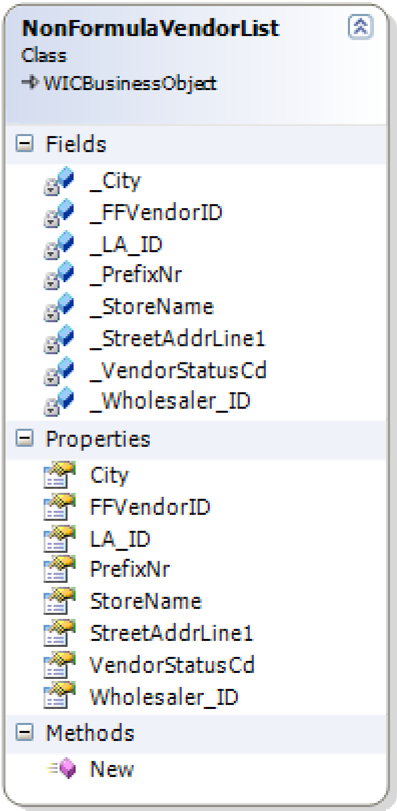
Methods
- New

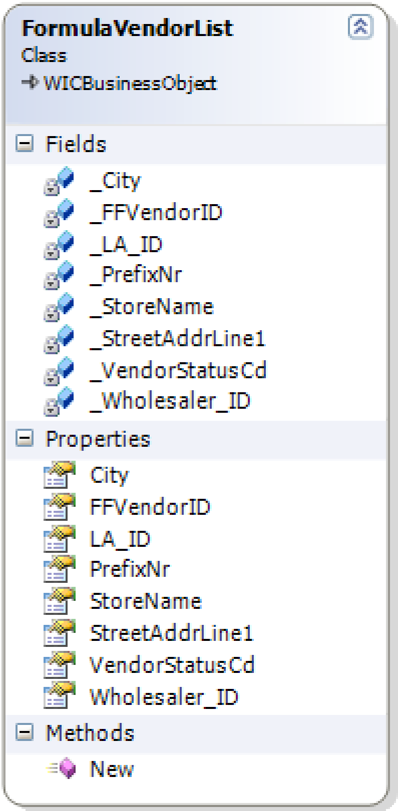
WholesalerAddress
Class
→ WICBusinessObject

Fields
- _AddressTypeCd
- _City
- _CountyNm
- _ID
- _InsertDt
- _InsertStfpID
- _ModifyDt
- _ModifyStfpID
- _POBox
- _State
- _StreetAddrLine1
- _StreetAddrLine2
- _Suite
- _Wholesaler_ID
- _ZipCode
- _ZipPlus4

Properties
- AddressTypeCd
- City
- CountyNm
- ID
- InsertDt
- InsertStfpID
- ModifyDt
- ModifyStfpID
- POBox
- State
- StreetAddrLine1
- StreetAddrLine2
- Suite
- Wholesaler_ID
- ZipCode
- ZipPlus4

Methods
- New


NonFormulaVendorList
Class
WICBusinessObject
Fields
_City
_FFVendorID
_LA_ID
_PrefixNr
_StoreName
_StreetAddrLine1
_VendorStatusCd
_Wholesaler_ID
Properties
City
FFVendorID
LA_ID
PrefixNr
StoreName
StreetAddrLine1
VendorStatusCd
Wholesaler_ID
Methods
New


FormulaVendorList
Class
WICBusinessObject
Fields
_City
_FFVendorID
_LA_ID
_PrefixNr
_StoreName
_StreetAddrLine1
_VendorStatusCd
_Wholesaler_ID
Properties
City
FFVendorID
LA_ID
PrefixNr
StoreName
StreetAddrLine1
VendorStatusCd
Wholesaler_ID
Methods
New